Automata completion and regularity preservation

Thomas Genet

IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France, `genet@irisa.fr`

Abstract

We consider rewriting of a regular language with a left-linear term rewriting system. We show two completeness theorems. The first one shows that, if the set of reachable terms is regular, then the equational tree automata completion can compute it. This was known to be true for some term rewriting system classes preserving regularity, but was still an open question in the general case. The proof is not constructive because it depends on regularity of the set of reachable terms, which is undecidable. The second theorem states that, if there exists a regular over-approximation of the set of reachable terms then completion can compute it (or safely under-approximate it). This theorem also provides an algorithmic way to safely explore regular approximations with completion. This has been implemented and used to verify safety properties, automatically, on first-order and higher-order functional programs. To carry out the proof, we also generalize and improve two results of completion: the Termination and the Upper-Bound theorems.

1998 ACM Subject Classification I.2.3 Deduction and Theorem Proving, F.4.2 Grammars and Other Rewriting Systems

Keywords and phrases term rewriting systems, regularity preservation, over-approximation, completeness, tree automata, tree automata completion

1 Introduction

Given a term rewriting system (TRS for short) $\mathcal{R}$ and a tree automaton $\mathcal{A}$ recognizing a regular tree language $\mathcal{L}(\mathcal{A})$, the set of reachable terms is $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) = \{t \mid s \in \mathcal{L}(\mathcal{A}) \text{ and } s \to_{\mathcal{R}}^* t\}$. In this paper, we show that the *equational tree automata completion algorithm* [16] is complete w.r.t. regular approximations. If $\mathcal{R}$ is left-linear and there exists a regular language $\mathscr{L}$ over-approximating $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$, i.e., $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathscr{L}$ then completion can build a tree automaton $\mathcal{B}$ such that $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{B}) \subseteq \mathscr{L}$. We also shows that completion is complete w.r.t. TRSs preserving regularity. If the regular language $\mathscr{L}$ is such that $\mathscr{L} = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ then completion can build a tree automaton $\mathcal{B}$ such that $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) = \mathcal{L}(\mathcal{B}) = \mathscr{L}$. On the one hand, automata built by completion-like algorithms are known to recognize *exactly* the set of reachable terms, for some *restricted* classes of TRSs [18, 24, 10, 12]. On the other hand, automata completion is able to build *over-approximations* for *any* left-linear TRS [11, 23, 16], and even for non-left-linear TRSs [2]. Such approximations are used for program verification [4, 3, 12, 14] as well as to automate termination proofs [17, 21]. To define approximations, completion uses an additional set of equations E and builds a tree automaton $\mathcal{A}^*_{\mathcal{R},E}$ such that $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. Until now it was an open question whether completion can build any regular over-approximation or compute the set of reachable terms if this set is regular. The first contribution of this paper is to answer this two questions in the positive, for left-linear TRSs. The proofs are not constructive because they rely on the assumption that a particular regular over-approximation exists or that $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ is regular, which is undecidable. For the approximated case, the proof is organized as follows. If there exists a regular over-approximation $\mathscr{L}$ such that $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathscr{L}$, we know that there exists a tree automaton $\mathcal{B}$ such that $\mathcal{L}(\mathcal{B}) = \mathscr{L}$. From $\mathcal{B}$, using the Myhill-Nerode theorem, we

can infer a set of equations E such that the set of E-equivalence classes $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite. Then we prove the following theorems:

(a) If $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite, then it is possible to build from E a set of equations E', equivalent to E, such that completion of any (reduced) automaton $\mathcal{A}$ by any TRS $\mathcal{R}$ with E' always terminates. This generalizes the termination theorem of [12];
(b) If $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite, then it is possible to build from E and $\mathcal{A}$ a tree automaton $\mathbb{A}$ recognizing the same language as $\mathcal{A}$ such that the completed automaton $\mathbb{A}^*_{\mathcal{R},E}$ has the following precision property: $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$, where $\mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$ is the set of reachable terms by rewriting modulo E. It generalizes the Upper Bound theorem of [16].
(c) Then, we show that $\mathcal{R}^*_E(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{B})$, and we get the main completeness theorem: $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathcal{R}^*_E(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{B})$.

Besides, we know from [16] that $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E})$. Thus, when using the set of equations defined from $\mathcal{B}$ to run completion, (c) implies that we can only get an approximation of $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ equivalent or better than $\mathscr{L} = \mathcal{L}(\mathcal{B})$. This result has a practical impact when approximations are used for software verification. In particular, for TRSs encoding functional programs, the search space of sets of equations E can be sufficiently constrained for enumeration to be possible. This has been implemented in the Timbuk [13] tool. The experiments show that this makes completion automatic enough to carry out efficiently safety proofs on first-order and higher-order functional programs. A corollary of (c) is another completeness result when $\mathscr{L}$ is *not* an approximation:

(d) If $\mathscr{L} = \mathcal{L}(\mathcal{B}) = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$, we can use $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E})$ to close-up the chain of $\subseteq$ and get that $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. Thus if $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ is regular, there exists a set of equations E s.t. $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$

Section 2 defines some basic notions in term rewriting and tree automata and Section 3 recalls the tree automata completion algorithm and the related theorems. Section 4 recalls the Myhill-Nerode theorem for trees and defines the functions to transform a set of equations into a tree automaton and vice versa. Section 5 proves Result (a) and Section 6 shows Result (b). Section 7 assembles (a) and (b) to prove results (c) and (d) using the proof sketched above. Section 8, shows how to take advantage of those results to program verification and presents some experiments. Finally, Section 9 concludes.

2 Preliminaries

In this section we introduce some definitions and concepts that will be used throughout the rest of the paper (see also [1, 6]). Let $\mathcal{F}$ be a finite set of symbols, each associated with an arity function. For brevity, we write $f : n$ if f is a symbol of arity n and $\mathcal{F}^n = \{f \in \mathcal{F} \mid f : n\}$. Let $\mathcal{X}$ be a countable set of *variables*, $\mathcal{T}(\mathcal{F}, \mathcal{X})$ denotes the set of *terms* and $\mathcal{T}(\mathcal{F})$ denotes the set of *ground terms* (terms without variables). The set of variables of a term t is denoted by $\mathscr{V}ar(t)$. A *substitution* is a function σ from $\mathcal{X}$ into $\mathcal{T}(\mathcal{F}, \mathcal{X})$, which can be uniquely extended to an endomorphism of $\mathcal{T}(\mathcal{F}, \mathcal{X})$. A *position* p in a term t is a finite word over $\mathbb{N}$, the set of natural numbers. The empty sequence λ denotes the top-most position. The set $\mathscr{P}os(t)$ of positions of a term t is inductively defined by $\mathscr{P}os(t) = \{\lambda\}$ if $t \in \mathcal{X}$ or t is a constant and $\mathscr{P}os(f(t_1, \ldots, t_n)) = \{\lambda\} \cup \{i.p \mid 1 \leq i \leq n \text{ and } p \in \mathscr{P}os(t_i)\}$ otherwise. If $p \in \mathscr{P}os(t)$, then $t(p)$ denotes the symbol at position p in t, $t|_p$ denotes the subterm of t at position p, and $t[s]_p$ denotes the term obtained by replacing the subterm $t|_p$ at position p by the term s. A ground context $C[\,]$ is a term in $\mathcal{T}(\mathcal{F} \cup \{\Box\})$ containing

exactly one occurrence of the symbol $\Box$. If $t \in \mathcal{T}(\mathcal{F})$ then $C[t]$ denotes the term obtained by the replacement of $\Box$ by t in $C[\,]$. A context is empty if it is equal to $\Box$. If $C[\,]$ is a context, then $C^1[\,] = C[\,]$ and for $n > 1$, $C^n[\,] = C[C^{n-1}[\,]]$.

A *term rewriting system* (TRS) $\mathcal{R}$ is a set of *rewrite rules* $l \to r$, where $l, r \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, $l \notin \mathcal{X}$, and $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$. A rewrite rule $l \to r$ is *left-linear* if each variable occurs only once in l. A TRS $\mathcal{R}$ is left-linear if every rewrite rule $l \to r$ of $\mathcal{R}$ is left-linear. The TRS $\mathcal{R}$ induces a rewriting relation $\to_{\mathcal{R}}$ on terms as follows. Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and $l \to r \in \mathcal{R}$, $s \to_{\mathcal{R}} t$ denotes that there exists a position $p \in \mathcal{P}os(s)$ and a substitution σ such that $s|_p = l\sigma$ and $t = s[r\sigma]_p$. The set of ground terms irreducible by a TRS $\mathcal{R}$ is denoted by $\textsc{Irr}(\mathcal{R})$ ($\textsc{Irr}(\mathcal{R}) \subseteq \mathcal{T}(\mathcal{F})$). A set $\mathcal{L} \subseteq \mathcal{T}(\mathcal{F})$ is $\mathcal{R}$-closed if for all $s \in \mathcal{L}$ and $s \to_{\mathcal{R}} t$ then $t \in \mathcal{L}$. The reflexive transitive closure of $\to_{\mathcal{R}}$ is denoted by $\to^*_{\mathcal{R}}$, and $s \to^!_{\mathcal{R}} t$ denotes that $s \to^*_{\mathcal{R}} t$ and t is irreducible by $\mathcal{R}$. The set of $\mathcal{R}$-descendants of a set of ground terms I is defined as $\mathcal{R}^*(I) = \{t \in \mathcal{T}(\mathcal{F}) \mid \exists s \in I \text{ s.t. } s \to^*_{\mathcal{R}} t\}$, i.e., the smallest $\mathcal{R}$-closed set containing I.

Let E be a *set of equations* $l = r$, where $l, r \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. The relation $=_E$ is the smallest congruence such that for all equations $l = r$ of E and for all substitutions σ we have $l\sigma =_E r\sigma$. The set of equivalence classes defined by $=_E$ on $\mathcal{T}(\mathcal{F})$ is denoted by $\mathcal{T}(\mathcal{F})/_{=_E}$. Given a TRS $\mathcal{R}$ and a set of equations E, a term $s \in \mathcal{T}(\mathcal{F})$ is rewritten modulo E into $t \in \mathcal{T}(\mathcal{F})$, denoted $s \to_{\mathcal{R}/E} t$, if there exist an $s' \in \mathcal{T}(\mathcal{F})$ and a $t' \in \mathcal{T}(\mathcal{F})$ such that $s =_E s' \to_{\mathcal{R}} t' =_E t$. The reflexive transitive closure $\to^*_{\mathcal{R}/E}$ of $\to_{\mathcal{R}/E}$ is defined as usual except that reflexivity is extended to terms equal modulo E, i.e., if for all $s, t \in \mathcal{T}(\mathcal{F})$, $s =_E t$ then $s \to^*_{\mathcal{R}/E} t$. The set of $\mathcal{R}$-descendants modulo E of a set of ground terms I is defined as $\mathcal{R}^*_E(I) = \{t \in \mathcal{T}(\mathcal{F}) \mid \exists s \in I \text{ s.t. } s \to^*_{\mathcal{R}/E} t\}$.

Let $\mathcal{Q}$ be a countably infinite set of symbols with arity 0, called *states*, such that $\mathcal{Q} \cap \mathcal{F} = \emptyset$. Terms in $\mathcal{T}(\mathcal{F} \cup \mathcal{Q})$ are called *configurations*. A *transition* is a rewrite rule $c \to q$, where c is a configuration and q is a state. A transition is *normalized* when $c = f(q_1, \ldots, q_n)$, $f \in \mathcal{F}$ is of arity n, and $q_1, \ldots, q_n \in \mathcal{Q}$. An *$\epsilon$-transition* is a transition of the form $q \to q'$ where q and q' are states. A bottom-up non-deterministic finite tree automaton (tree automaton for short) over the alphabet $\mathcal{F}$ is a tuple $\mathcal{A} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$, where $\mathcal{Q}_f \subseteq \mathcal{Q}$ is the set of final states, Δ is a finite set of normalized transitions and ϵ-transitions. An automaton is *epsilon-free* if it is free of ϵ-transitions. The transitive and reflexive *rewriting relation* on $\mathcal{T}(\mathcal{F} \cup \mathcal{Q})$ induced by the set of transitions Δ (resp. all transitions except ϵ-transitions) is denoted by $\to^*_{\Delta}$ (resp. $\to^{\setminus\epsilon *}_{\Delta}$). When Δ is attached to a tree automaton $\mathcal{A}$ we also denote those two relations by $\to_{\mathcal{A}}{}^*$ and $\to^{\setminus\epsilon *}_{\mathcal{A}}$, respectively. A tree automaton $\mathcal{A}$ is complete if for all $s \in \mathcal{T}(\mathcal{F})$ there exists a state q of $\mathcal{A}$ such that $s \to_{\mathcal{A}}{}^* q$. The *language* recognized by $\mathcal{A}$ in a state q is defined by $\mathcal{L}(\mathcal{A}, q) = \{t \in \mathcal{T}(\mathcal{F}) \mid t \to^*_{\mathcal{A}} q\}$. We define $\mathcal{L}(\mathcal{A}) = \bigcup_{q \in \mathcal{Q}_f} \mathcal{L}(\mathcal{A}, q)$. A state q of an automaton $\mathcal{A}$ is *reachable* if $\mathcal{L}(\mathcal{A}, q) \neq \emptyset$. An automaton is *reduced* if all its states are reachable. An automaton $\mathcal{A}$ is *$\setminus\epsilon$-reduced* if for all state q of $\mathcal{A}$ there exists a ground term $t \in \mathcal{T}(\mathcal{F})$ such that $t \to^{\setminus\epsilon *}_{\mathcal{A}} q$. An automaton $\mathcal{A}$ is deterministic if for all ground terms $s \in \mathcal{T}(\mathcal{F})$ and all states q, q' of $\mathcal{A}$, if $s \to_{\mathcal{A}}{}^* q$ and $s \to_{\mathcal{A}}{}^* q'$ then $q = q'$. An automaton $\mathcal{A}$ is $\mathcal{R}$-closed if for all terms s, t and all states $q \in \mathcal{Q}$, $s \to_{\mathcal{A}}{}^* q$ and $s \to_{\mathcal{R}} t$ implies $t \to_{\mathcal{A}}{}^* q$.

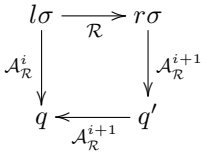
3 Equational Tree Automata Completion

Starting from a tree automaton $\mathcal{A}_0 = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta_0 \rangle$ and a left-linear TRS $\mathcal{R}$, the completion algorithm computes an automaton $\mathcal{A}^*$ such that $\mathcal{L}(\mathcal{A}^*) = \mathcal{R}^*(\mathcal{L}(\mathcal{A}_0))$ or $\mathcal{L}(\mathcal{A}^*) \supseteq$

$\mathcal{R}^*(\mathcal{L}(\mathcal{A}_0))$.

3.1 Completion General Principles

From $\mathcal{A}^0_\mathcal{R} = \mathcal{A}_0$, Tree automata completion successively computes tree automata $\mathcal{A}^1_\mathcal{R}$, $\mathcal{A}^2_\mathcal{R}$, ... such that for all $i \geq 0 : \mathcal{L}(\mathcal{A}^i_\mathcal{R}) \subseteq \mathcal{L}(\mathcal{A}^{i+1}_\mathcal{R})$ and if $s \in \mathcal{L}(\mathcal{A}^i_\mathcal{R})$, and $s \to_\mathcal{R} t$ then $t \in \mathcal{L}(\mathcal{A}^{i+1}_\mathcal{R})$. For $k \in \mathbb{N}$, if $\mathcal{L}(\mathcal{A}^k_\mathcal{R}) = \mathcal{L}(\mathcal{A}^{k+1}_\mathcal{R})$ then $\mathcal{A}^k_\mathcal{R}$ is a fixpoint and we denote it by $\mathcal{A}^*_\mathcal{R}$. To construct $\mathcal{A}^{i+1}_\mathcal{R}$ from $\mathcal{A}^i_\mathcal{R}$, we perform a *completion step* which consists in finding *critical pairs* between $\to_\mathcal{R}$ and $\to_{\mathcal{A}^i_\mathcal{R}}$. For a substitution $\sigma : \mathcal{X} \mapsto \mathcal{Q}$ and a rule $l \to r \in \mathcal{R}$, a critical pair is an instance $l\sigma$ of l such that there exists a state $q \in \mathcal{Q}$ satisfying $l\sigma \to^*_{\mathcal{A}^i_\mathcal{R}} q$ and $r\sigma \not\to^*_{\mathcal{A}^i_\mathcal{R}} q$. For $r\sigma$ to be recognized by the same state and thus model the rewriting of $l\sigma$ into $r\sigma$, it is enough to add the necessary transitions to $\mathcal{A}^i_\mathcal{R}$ in order to obtain $\mathcal{A}^{i+1}_\mathcal{R}$ such that $r\sigma \to^*_{\mathcal{A}^{i+1}_\mathcal{R}} q$. In [24, 16], critical pairs are joined in the following way:

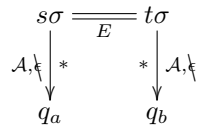


From an algorithmic point of view, there remain two problems to solve: find all the critical pairs $(l \to r, \sigma, q)$ and find the transitions to add to $\mathcal{A}^i_\mathcal{R}$ to have $r\sigma \to^*_{\mathcal{A}^{i+1}_\mathcal{R}} q$. The first problem, called matching, can be efficiently solved using a specific algorithm [10]. The second problem is solved using a normalization algorithm [12]. To have $r\sigma \to^*_{\mathcal{A}^{i+1}_\mathcal{R}} q'$ we need a transition of the form $r\sigma \to q'$ in $\mathcal{A}^{i+1}_\mathcal{R}$. However, it is possible that this transitions is not normalized. In this case, it is necessary to introduce new states and new transitions. For instance, to normalize a transition $f(g(a), h(q_1)) \to q'$ w.r.t. a tree automaton $\mathcal{A}^i_\mathcal{R}$ with transitions $a \to q_1$, $b \to q_1$, $g(q_1) \to q_1$, we first rewrite $f(g(a), h(q_1))$ with transitions of $\mathcal{A}^i_\mathcal{R}$ as far as possible. We obtain $f(q_1, h(q_1))$. Then we introduce the new state q_2 and the new transition $h(q_1) \to q_2$ to recognize the term $h(q_1)$. The new transitions to add to $\mathcal{A}^i_\mathcal{R}$ are thus: $h(q_1) \to q_2$, $f(q_1, q_2) \to q'$, and $q' \to q$.

3.2 Simplification of Tree Automata by Equations

Since completion creates new transitions and new states to join critical pairs, it may diverge. Divergence is avoided by *simplifying* the tree automaton with a set of equations E. This operation permits to over-approximate languages that cannot be recognized *exactly* using tree automata completion, e.g., non-regular languages. The simplification operation consists in finding E-equivalent terms recognized in $\mathcal{A}$ by different states and then by merging those states.

▶ **Definition 1** (Simplification relation)**.** Let $\mathcal{A} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be a tree automaton and E be a set of equations. For $s = t \in E$, $\sigma : \mathcal{X} \mapsto \mathcal{Q}$, $q_a, q_b \in \mathcal{Q}$ such that $s\sigma \to^{\not\in *}_\mathcal{A} q_a$, $t\sigma \to^{\not\in *}_\mathcal{A} q_b$, i.e.,



and $q_a \neq q_b$ then $\mathcal{A}$ is *simplified* into $\mathcal{A}'$, denoted by $\mathcal{A} \rightsquigarrow_E \mathcal{A}'$, where $\mathcal{A}'$ is $\mathcal{A}$ where q_b is replaced by q_a in $\mathcal{Q}$, $\mathcal{Q}_f$ and Δ. ◇

▶ **Example 2.** Let $E = \{s(s(x)) = s(x)\}$ and $\mathcal{A}$ be the tree automaton with $\mathcal{Q}_f = \{q_2\}$ and set of transitions $\Delta = \{a \to q_0, s(q_0) \to q_1, s(q_1) \to q_2\}$. Hence $\mathcal{L}(\mathcal{A}) = \{s(s(a))\}$. We can perform a simplification step using the equation $s(s(x)) = s(x)$ because we found a substitution $\sigma = \{x \mapsto q_0\}$ such that:

$$\begin{array}{ccc} s(s(q_0)) & \overset{E}{=\!=} & s(q_0) \\ {\scriptstyle \mathcal{A},\epsilon}\downarrow * & & * \downarrow{\scriptstyle \mathcal{A},\epsilon} \\ q_2 & & q_1 \end{array}$$

Hence, $\mathcal{A} \rightsquigarrow_E \mathcal{A}'$ where $\mathcal{A}'$ is $\mathcal{A}$ where q_2 is replaced by q_1 i.e., $\mathcal{A}'$ is the automaton with $\mathcal{Q}'_f = \{q_1\}$, $\Delta = \{a \to q_0, s(q_0) \to q_1, s(q_1) \to q_1\}$. Note that $\mathcal{L}(\mathcal{A}') = \{s^*(s(a))\}$.

The simplification relation $\rightsquigarrow_E$ is terminating and confluent (modulo state renaming) [16]. In the following, by $\mathcal{S}_E(\mathcal{A})$ we denote the unique automaton (modulo renaming) $\mathcal{A}'$ such that $\mathcal{A} \rightsquigarrow_E^* \mathcal{A}'$ and $\mathcal{A}'$ is irreducible (it cannot be simplified further).

3.3 The full Completion Algorithm

▶ **Definition 3** (Automaton completion)**.** Let $\mathcal{A}$ be a tree automaton, $\mathcal{R}$ a left-linear TRS and E a set of equations.

- $\mathcal{A}^0_{\mathcal{R},E} = \mathcal{A}$,
- $\mathcal{A}^{n+1}_{\mathcal{R},E} = \mathcal{S}_E(\mathcal{C}_\mathcal{R}(\mathcal{A}^n_{\mathcal{R},E}))$, for $n \geq 0$ where $\mathcal{C}_\mathcal{R}(\mathcal{A}^n_{\mathcal{R},E})$ is the tree automaton such that all critical pairs of $\mathcal{A}^n_{\mathcal{R},E}$ are joined.

If there exists $k \in \mathbb{N}$ such that $\mathcal{A}^k_{\mathcal{R},E} = \mathcal{A}^{k+1}_{\mathcal{R},E}$, then we write $\mathcal{A}^*_{\mathcal{R},E}$ for $\mathcal{A}^k_{\mathcal{R},E}$.

▶ **Example 4.** Let $\mathcal{R} = \{f(x,y) \to f(s(x),s(y))\}$, $E = \{s(s(x)) = s(x)\}$ and $\mathcal{A}^0$ be the tree automaton with set of transitions $\Delta = \{f(q_a,q_b) \to q_0, a \to q_a, b \to q_b\}$, i.e., $\mathcal{L}(\mathcal{A}^0) = \{f(a,b)\}$. The completion ends after two completion steps on $\mathcal{A}^2_{\mathcal{R},E}$ which is a fixpoint $\mathcal{A}^*_{\mathcal{R},E}$. Completion steps are summed up in the following table. To simplify the presentation, we do not repeat the common transitions: $\mathcal{A}^i_{\mathcal{R},E}$ and $\mathcal{C}_\mathcal{R}(\mathcal{A}^i)$ columns are supposed to contain all transitions of $\mathcal{A}^0, \ldots, \mathcal{A}^{i-1}_{\mathcal{R},E}$.

$\mathcal{A}^0$	$\mathcal{C}_\mathcal{R}(\mathcal{A}^0)$	$\mathcal{A}^1_{\mathcal{R},E}$	$\mathcal{C}_\mathcal{R}(\mathcal{A}^1_{\mathcal{R},E})$	$\mathcal{A}^2_{\mathcal{R},E}$
$f(q_a,q_b) \to q_0$	$f(q_1,q_2) \to q_3$	$f(q_1,q_2) \to q_3$	$f(q_4,q_5) \to q_6$	$f(q_1,q_2) \to q_6$
$a \to q_a$	$s(q_a) \to q_1$	$s(q_a) \to q_1$	$s(q_1) \to q_4$	$s(q_1) \to q_1$
$b \to q_b$	$s(q_b) \to q_2$	$s(q_b) \to q_2$	$s(q_2) \to q_5$	$s(q_2) \to q_2$
	$q_3 \to q_0$	$q_3 \to q_0$	$q_6 \to q_3$	

On $\mathcal{A}^0$, there is one critical pair $f(q_a,q_b) \to^*_{\mathcal{A}_0} q_0$ and $f(q_a,q_b) \to_\mathcal{R} f(s(q_a),s(q_b))$. The automaton $\mathcal{C}_\mathcal{R}(\mathcal{A}^0)$ contains all the transitions of $\mathcal{A}^0$ with the new transitions (and the new states) necessary to join the critical pair, i.e., to have $f(s(q_a),s(q_b)) \to^*_{\mathcal{C}_\mathcal{R}(\mathcal{A}^0)} q_0$. The automaton $\mathcal{A}^1_{\mathcal{R},E}$ is exactly $\mathcal{C}_\mathcal{R}(\mathcal{A}^0)$ because simplification by equations do not apply. Then, $\mathcal{C}_\mathcal{R}(\mathcal{A}^1_{\mathcal{R},E})$ contains all the transitions of $\mathcal{A}^1_{\mathcal{R},E}$ and $\mathcal{A}^0$ plus those obtained by the resolution of the critical pair $f(q_1,q_2) \to^*_{\mathcal{A}^1_{\mathcal{R},E}} q_3$ and $f(q_1,q_2) \to_\mathcal{R} f(s(q_1),s(q_2))$. On $\mathcal{C}_\mathcal{R}(\mathcal{A}^1_{\mathcal{R},E})$ simplification using the equation $s(s(x)) = s(x)$ can be applied on the following instances:

$s(s(q_a)) = s(q_a)$ and $s(s(q_b)) = q_b$. Since $s(s(q_a)) \to^*_{\mathcal{C}_\mathcal{R}(\mathcal{A}^1_{\mathcal{R},E})} q_4$ and $s(q_a) \to^*_{\mathcal{C}_\mathcal{R}(\mathcal{A}^1_{\mathcal{R},E})} q_1$, simplification merges q_4 with q_1. Similarly, simplification on $s(s(q_b)) = q_b$ merges q_5 with q_2. Thus, $\mathcal{A}^2_{\mathcal{R},E} = \mathcal{C}_\mathcal{R}(\mathcal{A}^1_{\mathcal{R},E})$ where q_4 is replaced by q_1 and q_5 is replaced q_2. This automaton is a fixed point because it has no other critical pairs (they are all joined).

3.4 Three Theorems on Completion

Tree automata completion enjoys three theorems defining its main properties. The first theorem is about *termination*. It defines a sufficient condition on E for completion to terminate. The second is a *sound approximation* theorem guaranteeing that completion always computes a tree automaton recognizing an over-approximation of reachable terms. This result is called the Lower Bound theorem. The third one, a *Precision* theorem, guarantees that the computed automaton recognizes only $\mathcal{R}/E$-reachable terms. This result is called the Upper Bound theorem. We first state the soundness theorem.

▶ **Theorem 5** (Lower Bound [16]). *Let $\mathcal{R}$ be a left-linear TRS, $\mathcal{A}$ be a tree automaton and E be a set of equations. If completion terminates on $\mathcal{A}^*_{\mathcal{R},E}$ then $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$.*

To state the upper bound theorem, we need the notion of $\mathcal{R}/E$-coherence we now define.

▶ **Definition 6** (Coherent automaton). Let $\mathcal{A} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be a tree automaton, $\mathcal{R}$ a TRS and E a set of equations. The automaton $\mathcal{A}$ is said to be *$\mathcal{R}/E$-coherent* if $\forall q \in \mathcal{Q}$: $\exists s \in \mathcal{T}(\mathcal{F})$:

$$s \to^{\not\epsilon *}_\mathcal{A} q \wedge [\forall t \in \mathcal{T}(\mathcal{F}) : (t \to^{\not\epsilon *}_\mathcal{A} q \implies s =_E t) \wedge (t \to_\mathcal{A}{}^* q \implies s \to^*_{\mathcal{R}/E} t)].$$

The intuition behind $\mathcal{R}/E$-coherence is the following. A $\mathcal{R}/E$-coherent is $\not\epsilon$-reduced, its ϵ-transitions represent rewriting steps and normalized transitions recognize E-equivalence classes. More precisely, in an $\mathcal{R}/E$-coherent tree automaton, if two terms s, t are recognized in the same state q using only normalized transitions then they belong to the same E-equivalence class. Otherwise, if at least one ϵ-transition is necessary to recognize, say, t in q then at least one step of rewriting was necessary to obtain t from s.

▶ **Example 7.** Let $\mathcal{R} = \{a \to b\}$, $E = \{c = d\}$ and $\mathcal{A} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ with $\Delta = \{a \to q_0, b \to q_1, c \to q_2, d \to q_2, q_1 \to q_0\}$. The automaton $\mathcal{A}$ is $\mathcal{R}/E$-coherent because it is $\not\epsilon$-reduced and the state q_2 recognizes with $\to^{\not\epsilon}_\Delta$ two terms c and d but they satisfy $c =_E d$. Finally, $a \to^*_\Delta q_0$ and $b \to^*_\Delta q_0$ but $a \to^{\not\epsilon}_\Delta q_0$, $b \to^{\not\epsilon}_\Delta q_1 \to q_0$ and $a \to_\mathcal{R} b$.

▶ **Theorem 8** (Upper Bound [16]). *Let $\mathcal{R}$ be a left-linear TRS, E a set of equations and $\mathcal{A}$ an $\mathcal{R}/E$-coherent automaton. For any $i \in \mathbb{N}$: $\mathcal{L}(\mathcal{A}^i_{\mathcal{R},E}) \subseteq \mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$ and $\mathcal{A}^i_{\mathcal{R},E}$ is $\mathcal{R}/E$-coherent.*

Finally, we state the termination theorem which relies on E-compatibility. Roughly speaking, E-compatibility is the symmetric of E-coherence. An automaton $\mathcal{A}$ is E-compatible if for all states $q_1, q_2 \in \mathcal{A}$ and all terms $s, t \in \mathcal{T}(\mathcal{F})$ such that $s \to^{\not\epsilon *}_\mathcal{A} q_1$, $t \to^{\not\epsilon *}_\mathcal{A} q_2$ and $s =_E t$ then we have $q_1 = q_2$.

▶ **Theorem 9** (Termination of completion [12]). *Let $\mathcal{A}$ be a $\not\epsilon$-reduced tree automaton, $\mathcal{R}$ a left-linear TRS, $j \in \mathbb{N}$, and E a set of equations such that $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite. If for all $i \geq j$, $\mathcal{A}^i_{\mathcal{R},E}$ is E-compatible then there exists a natural number n such that $\mathcal{A}^n_{\mathcal{R},E}$ is a fixpoint.*

To prove our final result, we first have to generalize Theorems 8 and 9 to discard the technical $\mathcal{R}/E$-coherence and E-compatibility assumptions. This is the objective of the next sections.

4 From automata to equations and vice versa

The above termination theorem uses the assumption that the automata $\mathcal{A}^i_{\mathcal{R},E}$ are all E-compatible. This assumption is not true in general and is not preserved by tree automaton completion: $\mathcal{A}^{i+1}_{\mathcal{R},E}$ may not be E-compatible even if $\mathcal{A}^i_{\mathcal{R},E}$ is.

▶ **Example 10.** Let $\mathcal{F} = \{f:1, a:0, b:0, c:0\}$, $\mathcal{R} = \{f(x) \to f(f(x)), f(f(x)) \to a\}$, $\mathcal{A}$ be the automaton such that $\Delta = \{a \to q_1, c \to q_1, f(q_1) \to q_f\}$ and $E = \{f(x) = b\}$. Note that $\mathcal{T}(\mathcal{F})/_{=_E}$ has 3 equivalence classes: the class of $\{a\}$, the class of $\{b, f(a), f(b), f(c), \ldots\}$ and the class of $\{c\}$. However, completion does not terminate on this example. Automaton $\mathcal{A}$ is E-compatible ($f(a) =_E f(c)$ and both terms are recognized by the same state: q_f) but $\mathcal{A}^1_{\mathcal{R},E}$ is not: it has one new state q_2 and contains additional transitions $\{f(q_f) \to q_2, q_2 \to q_f\}$. We thus have $f(f(a)) \to^*_{\mathcal{A}^1_{\mathcal{R},E}} q_2$ and $f(a) \to^*_{\mathcal{A}^1_{\mathcal{R},E}} q_f$ and $f(f(a)) =_E f(a)$ but $q_2 \neq q_f$. Since b is not recognized by $\mathcal{A}^n_{\mathcal{R},E}$ for any n, the equation $f(x) = b$ never applies and completion diverges.

Note that E-compatibility can be satisfied and preserved for particular cases of $\mathcal{R}$ and E, e.g., for typed functional programs [12]). Here, we show how to transform the set E into a set $E_{\mathcal{B}}$ for which completed automata are $E_{\mathcal{B}}$-compatible, and completion is thus terminating.[1] We also build $E_{\mathcal{B}}$ so that its precision is similar to E, i.e., $=_E \equiv =_{E_{\mathcal{B}}}$. This transformation is based on the Myhill-Nerode theorem for trees [19, 6]. We first produce a tree automaton $\mathcal{B}$ whose states recognize the equivalence classes of E. Then, from $\mathcal{B}$, we perform the inverse operation and obtain a set $E_{\mathcal{B}}$ whose set of equivalence classes is similar to the classes of E, but whose equations avoid the problem shown in Example 10. Since deciding finiteness of $\mathcal{T}(\mathcal{F})/_{=_E}$ and deciding $=_E$ is not always possible, we also propose an alternative version of this transformation using standard tools of rewriting: termination proofs and normalization. With this alternative transformation, $E_{\mathcal{B}}$ still ensures the termination of completion but can be more precise than E, i.e., $=_E \supseteq =_{E_{\mathcal{B}}}$.

4.1 From equations to automata

Provided that $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite, the Myhill-Nerode theorem for trees [19, 6] strongly relates $\mathcal{T}(\mathcal{F})/_{=_E}$ with tree automata. This theorem is constructive and provides an algorithm to switch from one form to the other, provided that $=_E$ is decidable. In the following we denote by $\mathbf{MN}$ the function that builds a tree automaton from a set of equations E, using the algorithm of [19].

▶ **Theorem 11** (Myhill-Nerode theorem for trees [19]). *If $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite and $=_E$ decidable, $\mathcal{B} = \mathbf{MN}(E)$ is a reduced, deterministic, epsilon-free and complete tree automaton such that for all $s, t \in \mathcal{T}(\mathcal{F})$, $s =_E t \iff (\exists q : \{s, t\} \subseteq \mathcal{L}(\mathcal{B}, q))$.*

However, determining whether $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite is not decidable in general. This is an unpublished result of S. Tison [25]. Since termination of the translation from E to $\mathcal{B}$ depends on the finiteness of $\mathcal{T}(\mathcal{F})/_{=_E}$, to use this algorithm we need, at least, a criterion for this property. Besides, the algorithm proposed in [19] needs $=_E$ to be decidable, which is not always true.

[1] We could also complete the automaton $\mathcal{A}^n_{\mathcal{R},E}$ with transitions recognizing the complement of $\mathcal{L}(\mathcal{A}^n_{\mathcal{R},E})$. All equations could be applied. Although it solves the termination problem, it may introduce additional approximations. In particular, the precision theorem of completion no longer holds.

Thus, we propose an alternative technique based on the TRS $\overrightarrow{E} = \{u \to v \mid u = v \in E\}$, where all equations $u = v$ are oriented so that $\overrightarrow{E}$ can be shown terminating. If we can orient E in $\overrightarrow{E}$ so that it is weakly terminating and $\text{IRR}(\overrightarrow{E})$ is finite then so is $\mathcal{T}(\mathcal{F})/_{=_E}$. This is due to the fact that $card(\mathcal{T}(\mathcal{F})/_{=_E}) \leq card(\text{IRR}(\overrightarrow{E}))$.[2] Note that the opposite is not true and, in particular, that $\text{IRR}(\overrightarrow{E})$ may be infinite though $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite.[3] Hopefully, finiteness of $\text{IRR}(\overrightarrow{E})$ is decidable [6]. Besides, we replace checking $s =_E t$ by a (weaker) test on normal forms of s and t. If $\overrightarrow{E}$ is weakly terminating, then we check if there exists a term u such that $s \to^!_{\overrightarrow{E}} u$, $t \to^!_{\overrightarrow{E}} u$. Note that, without confluence of $\overrightarrow{E}$, irreducible terms of $\text{IRR}(\overrightarrow{E})$ may not coincide with equivalence classes of $\mathcal{T}(\mathcal{F})/_{=_E}$. We will see in Section 5.2 that weak termination of $\overrightarrow{E}$ and finiteness of $\text{IRR}(\overrightarrow{E})$ are, in fact, sufficient to guarantee termination of completion. Now, we propose a function **E2A** which is a relaxed version of **MN** , i.e., $\mathbf{E2A}(\overrightarrow{E})$ can have more states than $\mathbf{MN}(E)$: $\mathbf{MN}(E)$ has $card(\mathcal{T}(\mathcal{F})/_{=_E})$ states and $\mathbf{E2A}(\overrightarrow{E})$ has $card(\text{IRR}(\overrightarrow{E}))$ states, where $card(\mathcal{T}(\mathcal{F})/_{=_E}) \leq card(\text{IRR}(\overrightarrow{E}))$, as shown above.

▶ **Definition 12** (Function **E2A**). Let $\overrightarrow{E}$ be a TRS, $\mathcal{Q}$ be a set of states and Δ be a set of transitions. Let $state : \mathcal{T}(\mathcal{F}) \mapsto \mathcal{Q}$ be an injective function mapping ground terms to state symbols. $\mathbf{E2A}(\overrightarrow{E}) = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}, \Delta \rangle$ where $\mathcal{Q} = \{state(u) \mid u \in \text{IRR}(\overrightarrow{E})\}$ and $\Delta = \{f(state(u_1), \ldots, state(u_n)) \to state(u) \mid u_1, \ldots, u_n, u \in \text{IRR}(\overrightarrow{E})$ and $f(u_1, \ldots, u_n) \to^!_{\overrightarrow{E}} u\}$

Note that **E2A** builds a finite automaton as soon as $\text{IRR}(\overrightarrow{E})$ is finite. Weak termination of $\overrightarrow{E}$ is enough because for each term s we only need one term t such that $s \to^!_{\overrightarrow{E}} t$. We assume that the strategy for rewriting s into t is known. Besides, for $\mathbf{E2A}(\overrightarrow{E})$ to be deterministic, we need the assumption that $\to^!_{\overrightarrow{E}}$ is, itself, deterministic. Thus, we assume that $\to^!_{\overrightarrow{E}}$ uses a deterministic strategy leading to irreducible terms. For instance, if $\overrightarrow{E}$ is innermost terminating, we can assume that $\to^!_{\overrightarrow{E}}$ uses leftmost innermost rewriting.

▶ **Example 13.** Consider the $E = \{f(x) = b\}$ of Example 10. Let us choose $\overrightarrow{E} = \{f(x) \to b\}$. Since $\overrightarrow{E}$ is left-linear, we can build a tree automaton recognizing $\text{IRR}(\overrightarrow{E})$ in an effective way [7, 5]. This tree automaton recognizes only 3 irreducible terms a, b, c. Furthermore $\overrightarrow{E}$ is terminating, we can thus build the automaton $\mathbf{E2A}(\overrightarrow{E})$. It has 3 states q_0, q_1, q_2 such that $state(a) = q_0$, $state(b) = q_1$ and $state(c) = q_2$. It has six transitions $a \to q_0$ (because $a \to^!_{\overrightarrow{E}} a$), $b \to q_1$ (because $b \to^!_{\overrightarrow{E}} b$), $c \to q_2$ (because $c \to^!_{\overrightarrow{E}} c$), $f(q_0) \to q_1$ (because $f(a) \to^!_{\overrightarrow{E}} b$), $f(q_1) \to q_1$ (because $f(b) \to^!_{\overrightarrow{E}} b$), $f(q_2) \to q_1$ (because $f(c) \to^!_{\overrightarrow{E}} b$).

The automaton $\mathbf{E2A}(\overrightarrow{E})$ enjoys properties close to the ones of $\mathbf{MN}(E)$.

▶ **Lemma 14.** *Let $\mathcal{B} = \mathbf{E2A}(\overrightarrow{E})$. For all states $q \in \mathcal{B}$ there exists an irreducible term $u \in \text{IRR}(\overrightarrow{E})$ such that $u \to^*_{\mathcal{B}} q$ and $state(u) = q$.*

Proof. For any state q, there exists an irreducible term u such that $q = state(u)$. Now, we prove by induction on the height of u that $u \to^*_{\mathcal{B}} q$. If u is a constant then, by definition

[2] For all terms $s \in \mathcal{T}(\mathcal{F})$, since $\overrightarrow{E}$ is weakly terminating, there exist a natural number k and a finite rewriting sequence $s \to_{\overrightarrow{E}} s_1 \to_{\overrightarrow{E}} \ldots \to_{\overrightarrow{E}} s_k$ such that $s_k \in \text{IRR}(\overrightarrow{E})$. Thus, we can build an equational derivation $s =_E s_1 =_E \ldots =_E s_k$. Since $s =_E s_k$ and $s_k \in \text{IRR}(\overrightarrow{E})$ there cannot be more than $card(\text{IRR}(\overrightarrow{E}))$ equivalence classes in $\mathcal{T}(\mathcal{F})/_{=_E}$.

[3] For instance, if $E = \{f(a) = b, a = b\}$ and $\overrightarrow{E} = \{f(a) \to b, a \to b\}$.

of $\mathcal{B}$, for all irreducible term t such that $u \to^!_{\overrightarrow{E}} t$, the transition $u \to state(t)$ belongs to $\mathcal{B}$. Since u is irreducible then $u = t$. If $u = f(u_1, \ldots, u_n)$, since u is irreducible then so are u_i, $1 \leq i \leq n$. Applying the induction hypothesis on the u_i's, we get that $u_i \to^*_{\mathcal{B}} q_i$ where $q_i = state(u_i)$ for $1 \leq i \leq n$. We conclude the proof by remarking that, by construction of $\mathcal{B}$, the transition $f(q_1, \ldots, q_n) \to q$ necessarily belongs to $\mathcal{B}$. ◀

▶ **Lemma 15.** *If* $\textsc{Irr}(\overrightarrow{E})$ *is finite,* $\overrightarrow{E}$ *weakly terminates, and* $\to^!_{\overrightarrow{E}}$ *is deterministic then* $\mathcal{B} = \mathbf{E2A}(\overrightarrow{E})$ *is a reduced, deterministic, epsilon-free and complete tree automaton.*

Proof. The first two assumptions are necessary to be sure that $\mathcal{B}$ exists. Automaton $\mathcal{B}$ is reduced because of Lemma 14. If $\to^!_{\overrightarrow{E}}$ is deterministic, then in the definition of Δ, there is only one possible term t s.t. $f(t_1, \ldots, t_n) \to^!_{\overrightarrow{E}} t$. Thus, for each configuration $f(q_1, \ldots, q_n)$ there is only one possible state q. This proves that $\mathcal{B}$ is deterministic. Automaton $\mathcal{B}$ is trivially epsilon-free. Finally, for completeness of $\mathcal{B}$, we can show that for all terms $s \in \mathcal{T}(\mathcal{F})$, there exists a state q s.t. $s \to^*_{\mathcal{B}} q$ by induction on the height of s. If s is a constant a, then by definition of $\mathcal{B}$, we know that there exists a transition $a \to state(t)$. For the inductive case, if $s = f(s_1, \ldots, s_n)$ then we know that there exists irreducible terms t_i and states q_i, $1 \leq i \leq n$ such that $s_i \to^*_{\mathcal{B}} q_i$. Then, by construction of $\mathcal{B}$, we know that there exists a transition $f(q_1, \ldots, q_n) \to q$, which concludes the proof. ◀

Theorem 11 tightly relates equivalence classes of $\mathcal{T}(\mathcal{F})/_{=_E}$ with languages recognized by $\mathbf{MN}(E)$. This relation also exists in $\mathbf{E2A}(\overrightarrow{E})$ but is slightly relaxed.

▶ **Lemma 16.** *Let* E *be a set of equations such that* $\textsc{Irr}(\overrightarrow{E})$ *is finite,* $\overrightarrow{E}$ *weakly terminates and* $\mathcal{B} = \mathbf{E2A}(\overrightarrow{E})$*. For all states* $q \in \mathcal{B}$*, for all terms* $s, t \in \mathcal{T}(\mathcal{F})$ *if* $s \to^*_{\mathcal{B}} q$ *and* $t \to^*_{\mathcal{B}} q$ *then* $s =_E t$*.*

Proof. First we prove that for all states $q \in \mathcal{B}$, for all terms $s \in \mathcal{T}(\mathcal{F})$ such that $s \to^*_{\mathcal{B}} q$, there exists an irreducible term u such that $q = state(u)$ and $s \to^!_{\overrightarrow{E}} u$. We prove this property by induction on the height of s. If s is a constant a, for $a \to^*_{\mathcal{B}} q$ to hold we know that there is necessarily an irreducible term u and a transition $a \to q$ in $\mathcal{B}$ such that $q = state(u)$ and $a \to^!_{\overrightarrow{E}} u$. For the inductive case, let $s = f(s_1, \ldots, s_n)$. Since $s \to^*_{\mathcal{B}} q$ we know that $s_i \to^*_{\mathcal{B}} q_i$ and there exists a transition $f(q_1, \ldots, q_n) \to q \in \mathcal{B}$. Applying the induction hypothesis on s_i, $1 \leq i \leq n$, we get that there exists irreducible terms u_i such that $q_i = state(u_i)$ and $s_i \to^!_{\overrightarrow{E}} u_i$ for $1 \leq i \leq n$. Thus $f(s_1, \ldots, s_n) \to^!_{\overrightarrow{E}} f(u_1, \ldots, u_n)$. Besides, for transition $f(q_1, \ldots, q_n) \to q$ to belong to $\mathcal{B}$, we know that there exists an irreducible term u such that $q = state(u)$ and $f(u_1, \ldots, u_n) \to^!_{\overrightarrow{E}} u$. We thus have $f(s_1, \ldots, s_n) \to^!_{\overrightarrow{E}} f(u_1, \ldots, u_n) \to^!_{\overrightarrow{E}} u$. This ends the proof by induction. Now, since $s \to^*_{\mathcal{B}} q$ and $t \to^*_{\mathcal{B}} q$, we know that there exists irreducible terms u and u' such that $q = state(u)$, $s \to^!_{\overrightarrow{E}} u$, $q = state(u')$, $t \to^!_{\overrightarrow{E}} u'$. Since *state* is an injective function, we get $u = u'$. Finally, $s \to^!_{\overrightarrow{E}} u$, $t \to^!_{\overrightarrow{E}} u$ implies $s =_E u =_E t$. ◀

4.2 From automata to equations

In the other direction, starting from a tree automaton $\mathcal{B}$ it is possible to build a set of equations $E_{\mathcal{B}}$ such that languages recognized by states of $\mathcal{B}$ and equivalence classes of $\mathcal{T}(\mathcal{F})/_{=_{E_{\mathcal{B}}}}$ coincide. This is the function **A2E** that is also described in [19]. We reformulate this function because we need some additional properties on the generated set of equations for completion to terminate. For simplicity we assume that $\mathcal{B}$ is **R**educed and epsilon-**F**ree.

Some properties of $E_{\mathcal{B}}$ will hold only if $\mathcal{B}$ is also **C**omplete and **D**eterministic. In the following, we use the **RF** and **RDFC** short-hands for automata having the related properties. Recall that for any tree automaton, there exists an equivalent **RF** or **RDFC** automaton [6].

For **RF** automata, the construction of $E_{\mathcal{B}}$ is straightforward and follows [19]: for all states q we identify a ground term recognized by q, a representative, and for all transitions $f(q_1, \ldots, q_n) \to q$ we generate an equation $f(t_1, \ldots, t_n) = t$ where t_i, $1 \leq i \leq n$ are representatives for q_i and t is a representative for q. However, for this set of equations to guarantee termination of completion it needs some redundancy: for each state we generate a *set* of *state representatives* and the equations are defined for each representative of the set. As shown in Example 10, the equation $f(x) = b$ cannot be applied during completion because b does not occur in the tree automaton. However, a logic consequence of this equation is that $f(f(a)) =_E f(a)$ and terms $f(f(a))$ and $f(a)$ that occur in the tree automaton could be merged. In our setting the term $f(a)$ will be a state representative and the equation $f(f(a)) = f(a)$ will appear in the set of generated equations. Roughly speaking, every constant symbol a appearing in a transition $a \to q$ is a state representative for q. Every term of the form $f(u_1, \ldots, u_n)$ is a state representative for q if (1) u_i's are not state representatives of q, (2) $f(q_1, \ldots, q_n) \to q$ is a transition of $\mathcal{B}$ and (3) u_i's are state representatives for the q_i's. The property (1) ensures finiteness of the set of representatives.

▶ **Definition 17** (State representatives)**.** Let $\mathcal{B} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be an **RF** tree automaton and $q \in \mathcal{Q}$. The set of state representatives of q of height lesser or equal to n, denoted by $[\![q]\!]^n_{\mathcal{B}}$, is inductively defined by:

- $[\![q]\!]^1_{\mathcal{B}} = \{a \mid a \to q \in \Delta\}$
- $[\![q]\!]^n_{\mathcal{B}} = [\![q]\!]^{n-1}_{\mathcal{B}} \cup \{f(u_1, \ldots, u_n) \mid f(q_1, \ldots, q_n) \to q \in \Delta \text{ and } \forall i \in \{1, \ldots, n\} : u_i \in [\![q_i]\!]_{\mathcal{B}},$ and $\forall p \in \mathscr{P}os(u_i) : u_i|_p \notin [\![q]\!]^{n-1}_{\mathcal{B}}\}$

In the above definition, the fact that $\mathcal{B}$ is reduced and epsilon-free ensures that there exists at least one (non-epsilon) transition for every state and that each state has at least one state representative.

▶ **Example 18.** Let $\mathcal{B}$ be the **RF** automaton that we obtained in Example 13 and whose set of transitions is $a \to q_0$, $b \to q_1$, $c \to q_2$, $f(q_0) \to q_1$, $f(q_1) \to q_1$, $f(q_2) \to q_1$.

- $[\![q_0]\!]^1_{\mathcal{B}} = \{a\}$, $[\![q_1]\!]^1_{\mathcal{B}} = \{b\}$, and $[\![q_2]\!]^1_{\mathcal{B}} = \{c\}$.
- $[\![q_0]\!]^2_{\mathcal{B}} = [\![q_0]\!]^1_{\mathcal{B}}$, $[\![q_1]\!]^2_{\mathcal{B}} = \{b, f(a), f(c)\}$, and $[\![q_2]\!]^2_{\mathcal{B}} = [\![q_2]\!]^1_{\mathcal{B}}$. The term $f(b)$ of height 2 and recognized by q_1 is not added to $[\![q_1]\!]^2_{\mathcal{B}}$ because its subterm b belongs to $[\![q_1]\!]^1_{\mathcal{B}}$.
- The fixpoint is reached because terms $f(f(a))$ and $f(f(c))$ recognized by q_1 are not added to $[\![q_1]\!]^3_{\mathcal{B}}$ because $f(a)$ and $f(c)$ belong to $[\![q_1]\!]^2_{\mathcal{B}}$.

We denote by $[\![q]\!]_{\mathcal{B}}$ the set of all state representatives for the state q i.e., the fixpoint of the above equations. Now, we show that for all reduced and epsilon-free automata, such a fixpoint exists and is always a finite set.

▶ **Lemma 19** (The set of state representatives is finite)**.** *For all* **RF** *tree automata $\mathcal{B}$, for all state $q \in \mathcal{B}$ there exists a natural number n for which the set $[\![q]\!]^n_{\mathcal{B}}$ is a fixpoint.*

Proof. We make a proof by contradiction. Assume that one set of state representatives $[\![q]\!]_{\mathcal{B}}$ is infinite. Let $\mathcal{Q}$ be the set of states of $\mathcal{B}$ and $t \in [\![q]\!]_{\mathcal{B}}$ be a term s.t. $|t| > \mathcal{C}\text{ard}(\mathcal{Q})$. Assume that we label each subterm of t by the state recognizing it in $\mathcal{B}$. Since height of t is greater than $\mathcal{C}\text{ard}(\mathcal{Q})$, by the pigeonhole principle we know that there exists $q' \in \mathcal{B}$ and a path in the tree t such that q' appears at least two times. Let $p, r \in \mathscr{P}os(t)$ be the positions

of the two subterms recognized by q'. By definition of state representatives, we know that $t|_p \in [\![q']\!]_{\mathcal{B}}$ and $t|_r \in [\![q']\!]_{\mathcal{B}}$. Since p and r are on the same path, we know that $t|_p$ is a strict subterm of $t|_r$ (or the opposite). This contradicts Definition 17 that forbids a term and a strict subterm to belong to the same set of representatives.

◀

▶ **Definition 20** (Function **A2E**: set of equations $E_{\mathcal{B}}$ from a tree automaton $\mathcal{B}$)**.** Let $\mathcal{B} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be an **RF** tree automaton. The set of equations $E_{\mathcal{B}}$ inferred from $\mathcal{B}$ is $\mathbf{A2E}(\mathcal{B}) = E_{\mathcal{B}} = \{f(u_1, \ldots, u_n) = u \mid f(q_1, \ldots, q_n) \to q \in \mathcal{B},\ u \in [\![q]\!]_{\mathcal{B}} \text{ and } u_i \in [\![q_i]\!]_{\mathcal{B}} \text{ for } 1 \leq i \leq n\}$.

▶ **Example 21.** Starting from the automaton $\mathcal{B}$ and the state representatives of Example 18, the set $\mathbf{A2E}(\mathcal{B})$ contains the following equations: $a = a$ (because of transition $a \to q_0$), $c = c$ (because of transition $c \to q_2$), $b = b$, $b = f(a)$, $b = f(c)$ (because of transition $b \to q_1$), $f(a) = f(a)$, $f(a) = b$, $f(a) = f(c)$ (because of transition $f(q_0) \to q_1$), $f(f(a)) = f(a)$, $f(f(a)) = b$, $f(f(a)) = f(c)$, $f(b) = f(a)$, $f(b) = b$, $f(b) = f(c)$, $f(f(c)) = f(a)$, $f(f(c)) = b$, $f(f(c)) = f(c)$ (because of transition $f(q_1) \to q_1$), $f(c) = f(a)$, $f(c) = b$, and $f(c) = f(c)$ (because of transition $f(q_2) \to q_1$).

Since $\mathcal{B}$ is finite, since the set of state representatives is finite, then so is $E_{\mathcal{B}}$. Note that many equations of $E_{\mathcal{B}}$ are useless w.r.t. the underlying equational theory. This is the case, in the above example, for equations of the form $a = a$ as well as the equation $f(a) = f(c)$ which is redundant w.r.t. $b = f(a)$ and $b = f(c)$. However, as shown in Example 10 those equations are necessary for equational simplification to produce $E_{\mathcal{B}}$-compatible automata and completion to terminate. With the above $E_{\mathcal{B}}$, completion of Example 10 terminates. Below, Theorem 26 shows that, if $\mathcal{B}$ is **RDFC** then completion with $\mathbf{A2E}(\mathcal{B})$ always terminates. Unsurprisingly, if $\mathcal{B}$ is deterministic then equivalence classes of $E_{\mathcal{B}}$ coincide with languages recognized by states of $\mathcal{B}$. This is the purpose of the next two lemmas.

▶ **Lemma 22.** *Let $\mathcal{B} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be an* **RDFC** *tree automaton and $E_{\mathcal{B}} = \mathbf{A2E}(\mathcal{B})$. For all $s \in \mathcal{T}(\mathcal{F})$, there exists a unique state $q \in \mathcal{Q}$ such that $s \to^*_{\mathcal{B}} q$ and for all state representatives $u \in [\![q]\!]_{\mathcal{B}}$, $s =_{E_{\mathcal{B}}} u$.*

Proof. We make a proof by induction on the height of s. If s is a constant, since $\mathcal{B}$ is complete and deterministic there exists a unique transition $s \to q \in \Delta$. By construction of $E_{\mathcal{B}}$, we know that there are equations with s on the left-hand side and all state representatives of $[\![q]\!]_{\mathcal{B}}$ on the right-hand side. For all equation $s = u$ with $u \in [\![q]\!]_{\mathcal{B}}$ we thus trivially have $s =_{E_{\mathcal{B}}} u$. This concludes the base case.

Now, we assume that the property is true for terms of height lesser or equal to n. Let $s = f(t_1, \ldots, t_n)$ where $t_1, \ldots, t_n$ are terms of height lesser or equal to n. Since $\mathcal{B}$ is complete, we know that there exists a state q such that $f(t_1, \ldots, t_n) \to^*_{\mathcal{B}} q$, i.e., there exists states $q_1, \ldots, q_n$ such that $f(q_1, \ldots, q_n) \to q \in \Delta$ and $t_i \to^*_{\mathcal{B}} q_i$ for $1 \leq i \leq n$. Using the induction hypothesis we get that there exist states q'_i in $\mathcal{B}$ and terms $[\![q'_i]\!]_{\mathcal{B}}$ such that $t_i \to^*_{\mathcal{B}} q_i$ and $t_i =_{E_{\mathcal{B}}} u_i$ for $u_i \in [\![q'_i]\!]_{\mathcal{B}}$ and for $1 \leq i \leq n$. Since $\mathcal{B}$ is deterministic, from $t_i \to^*_{\mathcal{B}} q_i$ and $t_i \to^*_{\mathcal{B}} q'_i$ we get that $q_i = q'_i$ and thus $t_i =_{E_{\mathcal{B}}} u_i$ for $u_i \in [\![q_i]\!]_{\mathcal{B}}$, with $1 \leq i \leq n$. Besides, since $f(q_1, \ldots, q_n) \to q \in \Delta$, we know that $E_{\mathcal{B}}$ contains the equations $f(u_1, \ldots, u_n) = u$ for all $u_i \in [\![q_i]\!]_{\mathcal{B}}$, for all $1 \leq i \leq n$ and for all $u \in [\![q]\!]_{\mathcal{B}}$. Thus q is the unique state such that $f(t_1, \ldots, t_n) \to^*_{\mathcal{B}} q$. Furthermore, $f(t_1, \ldots, t_n) =_{E_{\mathcal{B}}} f(u_1, \ldots, u_n) =_{E_{\mathcal{B}}} u$ for all $u_i \in [\![q_i]\!]_{\mathcal{B}}$, for all $1 \leq i \leq n$ and for all $u \in [\![q]\!]_{\mathcal{B}}$. ◀

Now we can relate equivalence classes of $E_{\mathcal{B}}$ and languages recognized by states of $\mathcal{B}$.

▶ **Lemma 23** (Equivalence classes of $E_\mathcal{B}$ coincide with languages recognized by states of $\mathcal{B}$). *Let $\mathcal{B} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be an* **RDFC** *tree automaton and $E_\mathcal{B} = \mathbf{A2E}(\mathcal{B})$. For all $s, t \in \mathcal{T}(\mathcal{F})$, $s =_{E_\mathcal{B}} t \iff (\exists q : \{s, t\} \subseteq \mathcal{L}(\mathcal{B}, q))$.*

Proof. For s and t, using Lemma 22, we know that there exist unique states $q, q' \in \mathcal{Q}$ such that $s \to^*_\mathcal{B} q$, $t \to^*_\mathcal{B} q'$ and for all state representatives $u \in [\![q]\!]_\mathcal{B}$ and $v \in [\![q']\!]_\mathcal{B}$, we have $s =_{E_\mathcal{B}} u$ and $t =_{E_\mathcal{B}} v$. We first prove the left to right implication. From $s =_{E_\mathcal{B}} t$ we obtain that $u =_{E_\mathcal{B}} v$, where u and v are state representatives. By construction of term representatives, for all states q we know that $[\![q]\!]_\mathcal{B}$ only contains terms recognized by q in $\mathcal{B}$. Since $\mathcal{B}$ is deterministic, if $q \neq q'$ then we can conclude that $[\![q]\!]_\mathcal{B} \cap [\![q']\!]_\mathcal{B} = \emptyset$. Thus, the only possibility to have $u =_{E_\mathcal{B}} v$ is to have an equation $u = v$ in $E_\mathcal{B}$. This entails that u and v belong to the same set of representatives: $[\![q]\!]_\mathcal{B} = [\![q']\!]_\mathcal{B}$, which entails that $q = q'$. Then $s \to^*_\mathcal{B} q$ and $t \to^*_\mathcal{B} q$ entails that $\{s, t\} \subseteq \mathcal{L}(\mathcal{B}, q)$. To prove the right to left implication, it is enough to point out that because of the determinism of $\mathcal{B}$ having $t \to^*_\mathcal{B} q'$ (the initial assumption) and having $t \to^*_\mathcal{B} q$ (the fact that $t \in \mathcal{L}(\mathcal{B}, q)$) is possible only if $q = q'$. This entails that u and v have a common set of representatives and thus for all representatives u of this set $s =_{E_\mathcal{B}} u =_{E_\mathcal{B}} t$. ◀

▶ **Corollary 24** ($\mathcal{T}(\mathcal{F})/_{=_{E_\mathcal{B}}}$ is finite). *Let $\mathcal{B} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be an* **RDFC** *tree automaton. If $E_\mathcal{B}$ is the set of equations inferred from $\mathcal{B}$ then $\mathcal{T}(\mathcal{F})/_{=_{E_\mathcal{B}}}$ is finite.*

Proof. Using Lemma 22, we know that for all terms $t \in \mathcal{T}(\mathcal{F})$ there exist a state $q \in \mathcal{Q}$ and a state representative $u \in [\![q]\!]_\mathcal{B}$ such that $t \to^*_\mathcal{B} q$ and $t =_{E_\mathcal{B}} u$. Since the number of states of $\mathcal{B}$ is finite, and since the set of state representatives u is finite for all states of $\mathcal{B}$ (Lemma 19), so is the number of equivalence classes of $\mathcal{T}(\mathcal{F})/_{=_{E_\mathcal{B}}}$. ◀

5 Generalizing the termination theorem

Now, we prove that using $E_\mathcal{B}$ built from an **RDFC** tree automaton $\mathcal{B}$, completion terminates.

5.1 Proving termination with $E_\mathcal{B}$

To prove this result, we need to show several results on the limit automaton of completion. In the following, the automaton $\mathcal{A}^*$ is the limit of the (possibly) infinite completion of an initial $\not\in$-reduced tree automaton $\mathcal{A}$ with $\mathcal{R}$ and $E_\mathcal{B}$. If the initial automaton is not $\not\in$-reduced then completion may diverge. For instance, completion of the automaton whose set of transitions is $\{f(q_0) \to q_1\}$, with $\mathcal{R} = \{f(x) \to f(f(x))\}$ and $E = \{f(a) = a\}$ diverges (simplification never happens because q_0 does not recognize any term). Now we show that all state representatives are recognized by epsilon-free derivations in $\mathcal{A}^*$.

▶ **Lemma 25** (All states of $\mathcal{A}^*$ recognize at least one state representative). *Let $\mathcal{R}$ be a TRS, $\mathcal{A}$ a $\not\in$-reduced tree automaton, $\mathcal{B}$ an* **RDFC** *tree automaton and $E_\mathcal{B} = \mathbf{A2E}(\mathcal{B})$. Let $\mathcal{A}^*$ be the limit of the completion of $\mathcal{A}$ by $\mathcal{R}$ and $E_\mathcal{B}$. For all states $q \in \mathcal{A}^*$, for all terms $s \in \mathcal{T}(\mathcal{F})$ such that $s \to^{\not\in *}_{\mathcal{A}^*} q$, there exists a state $q'_\mathcal{B} \in \mathcal{B}$, a term $u \in [\![q'_\mathcal{B}]\!]_\mathcal{B}$ such that $u =_{E_\mathcal{B}} s$ and $u \to^{\not\in *}_{\mathcal{A}^*} q$.*

Proof. Note that if $\mathcal{A}$ is $\not\in$-reduced, then so is $\mathcal{A}^*$ (cf. Lemma 44 of [12]). This is easy to figure out since all states added during completion recognize at least one term with $\to^{\not\in *}_{\mathcal{A}}$, and this is trivially preserved by simplification. By induction on the height of s we show that the representative u exists and is recognized by q. If s is of height 1 (it is a constant)

then, by construction of state representatives, we know that s is a representative. Thus $s = u \to^{*}_{\mathcal{A}^*} q$.

For the inductive case, assume that the property is true for all terms of height lesser or equal to n. Let $s = f(s_1, \ldots, s_n)$ be a term of height $n+1$. By assumption, we know that $f(s_1, \ldots, s_n) \to^{*}_{\mathcal{A}^*} q$. From $f(s_1, \ldots, s_n) \to^{*}_{\mathcal{A}^*} q$, we obtain that there exists states $q_1, \ldots, q_n$ of $\mathcal{A}^*$ such that $s_i \to^{*}_{\mathcal{A}^*} q_i$ for $i = 1, \ldots, n$ and a transition $f(q_1, \ldots, q_n) \to q$ in $\mathcal{A}^*$. Using the induction hypothesis on q_i, $i = 1, \ldots, n$ we get that there exist state representatives u_i such that $s_i =_{E_\mathcal{B}} u_i$ and $u_i \to^{*}_{\mathcal{A}^*} q_i$ for $i = 1, \ldots, n$. Then, since $f(q_1, \ldots, q_n) \to q$ in $\mathcal{A}^*$ we know that $f(u_1, \ldots, u_n) \to^{*}_{\mathcal{A}^*} q$. If $f(u_1, \ldots, u_n)$ is a state representative we are done since $f(s_1, \ldots, s_n) =_{E_\mathcal{B}} f(u_1, \ldots, u_n)$ and $f(u_1, \ldots, u_n) \to^{*}_{\mathcal{A}^*} q$. Otherwise, by definition of state representatives, for $u = f(u_1, \ldots, u_n)$ not to belong to the representatives there is a position p in u, different from the root position such that the subterm $u|_p$ is itself a state representative and it belongs to the same class as u, i.e., $u =_{E_\mathcal{B}} u|_p$. Since $u_1, \ldots, u_n$ are state representatives and $f(u_1, \ldots, u_n)$ is in the same equivalence class as $u|_p$ which is a state representative, we know that the equation $f(u_1, \ldots, u_n) = u|_p$ necessarily belongs to $E_\mathcal{B}$. Besides, for $u \to^{*}_{\mathcal{A}^*} q$ to hold, we know that there exists a state q' such that $u[u|_p]_p \to^{*}_{\mathcal{A}^*} u[q']_p \to^{*}_{\mathcal{A}^*} q$. Thus, $f(u_1, \ldots, u_n) \to^{*}_{\mathcal{A}^*} q$ and $u|_p \to^{*}_{\mathcal{A}^*} q'$. Then, since $E_\mathcal{B}$ contains the equation $f(u_1, \ldots, u_n) = u|_p$, and since $\mathcal{A}^*$ is simplified w.r.t. $E_\mathcal{B}$, we necessarily have $q = q'$ in $\mathcal{A}^*$. Finally, we have $f(s_1, \ldots, s_n) =_{E_\mathcal{B}} f(u_1, \ldots, u_n) =_{E_\mathcal{B}} u|_p$ and $u|_p \to^{*}_{\mathcal{A}^*} q$ where $u|_p$ is a state representative. ◀

Now, we can state the termination Theorem with $E_\mathcal{B}$.

▶ **Theorem 26** (Completion with $E_\mathcal{B}$ terminates)**.** *Let $\mathcal{R}$ be a TRS, $\mathcal{A}$ a $\backslash$-reduced tree automaton, $\mathcal{B}$ be an* **RDFC** *tree automaton and $E_\mathcal{B} =$ **A2E**$(\mathcal{B})$. Let n be the number of all states representatives of $\mathcal{B}$. The automaton $\mathcal{A}^*$, limit of the completion of $\mathcal{A}$ with $\mathcal{R}$ and $E_\mathcal{B}$, has n states or less.*

Proof. Recall that the number n of state representatives is finite (cf. Lemma 19). Assume that $\mathcal{A}^*$ has m distinct states with $m > n$. From Lemma 25 we know that for all state $q \in \mathcal{A}^*$, there exists a state representative u such that $u \to^{*}_{\mathcal{A}^*} q$. Since there are only n state representatives, by pigeon hole principle, we know that there is necessarily one state representative u recognized by two distinct states q_1 and q_2 of $\mathcal{A}^*$. Thus, $u \to^{*}_{\mathcal{A}^*} q_1$ and $u \to^{*}_{\mathcal{A}^*} q_2$. Besides, by construction of $E_\mathcal{B}$, we know that the equation $u = u$ is part of $E_\mathcal{B}$. This contradicts the fact that $\mathcal{A}^*$ is simplified w.r.t. $E_\mathcal{B}$. ◀

5.2 Building $E_\mathcal{B}$ from any set of equations E

Now, we combine the transformations **A2E** and **E2A** (or **A2E** and **MN**) to produce a set of equations $E_\mathcal{B}$ (from E) that ensures termination of completion and that is equivalent to E. We first prove that $E_\mathcal{B}$ is at least as precise as E.

▶ **Lemma 27.** *Let E be a set of equations. If $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite and $=_E$ is decidable then $E_\mathcal{B} =$ **A2E**$($**MN**$(E))$ and $=_E \equiv =_{E_\mathcal{B}}$.*

Proof. Let $\mathcal{B} =$ **MN**$(\overrightarrow{E})$. From Lemma 15, we know that $\mathcal{B}$ is **RDFC** and languages recognized by states of $\mathcal{B}$ coincide with equivalence classes of E. Then, let $E_\mathcal{B} =$ **A2E**$(\mathcal{B})$. Using Lemma 23, we get that languages of $\mathcal{B}$ coincide with equivalence classes of $E_\mathcal{B}$. Thus, $=_E \equiv =_{E_\mathcal{B}}$. ◀

▶ **Lemma 28.** *Let E be a set of equations. If* $\textsc{Irr}(\overrightarrow{E})$ *is finite and* $\overrightarrow{E}$ *weakly terminating then* $E_{\mathcal{B}} = \mathbf{A2E}(\mathbf{E2A}(\overrightarrow{E}))$ *and* $=_E \supseteq =_{E_{\mathcal{B}}}$.

Proof. Assume that $s =_{E_{\mathcal{B}}} t$. From Lemma 23, we get that there exists a state q in $\mathcal{B}$ such that $s \to^*_{\mathcal{B}} q$ and $t \to^*_{\mathcal{B}} q$. Then, with Lemma 16, we get that $s =_E t$. ◀

▶ **Theorem 29** (Generalized termination theorem for completion)**.** *Let E be a set of equations such that $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite and $=_E$ is decidable (resp.* $\textsc{Irr}(\overrightarrow{E})$ *is finite and* $\overrightarrow{E}$ *weakly terminating). For all $\not\in$-reduced tree automata $\mathcal{A}$ and TRSs $\mathcal{R}$, completion of $\mathcal{A}$ with $\mathcal{R}$ and* $\mathbf{A2E}(\mathbf{MN}(E))$ *(resp.* $\mathbf{A2E}(\mathbf{E2A}(\overrightarrow{E}))$*) terminates.*

Proof. Let $\mathcal{B} = \mathbf{MN}(E)$ (resp. $\mathcal{B} = \mathbf{E2A}(\overrightarrow{E})$). Using Lemma 15 (resp. Lemma 14), we know that $\mathcal{B}$ is **RDFC**. Let $E_{\mathcal{B}}$ be the set of equations $\mathbf{A2E}(\mathcal{B})$. Using Theorem 26, we know that completion of $\mathcal{A}$ with $\mathcal{R}$ and $E_{\mathcal{B}}$ is terminating. ◀

The above theorem shows how to tune a set of equations E into $E_{\mathcal{B}}$ to guarantee termination of completion. Note that tuning E into $E_{\mathcal{B}}$ does not jeopardize the precision of the completion since Lemma 28 guarantees that $=_E \supseteq =_{E_{\mathcal{B}}}$. This Lemma combined with Theorem 8 (the Upper Bound Theorem) ensures that completion with $E_{\mathcal{B}}$ can only be more precise than completion with E. In the next section, we improve the precision theorem itself.

6 Improving the Precision of Equational completion

Looking at our overall goal, we are half way there. If $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ is regular then it can be recognized by an automaton $\mathcal{B}$ and, using the results of the last section, we can build a set of equations $E_{\mathcal{B}}$ guaranteeing termination of completion. What remains to be proved is that completion with $E_{\mathcal{B}}$ ends on a tree automaton recognizing exactly $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. As it is, Theorem 8 (the Upper Bound Theorem) fails to tackle this goal because it needs $\mathcal{R}/E$-coherence of $\mathcal{A}$. However, if $\mathcal{A}$ is not $\mathcal{R}/E$-coherent the full precision, granted by this theorem, cannot be obtained.

▶ **Example 30.** Starting from Example 10, together with the set of equations $E_{\mathcal{B}}$ of Example 21, the initial tree automaton is not $\mathcal{R}/E_{\mathcal{B}}$-coherent (nor $\mathcal{R}/E$-coherent): $a \to^{\not\in *}_{\mathcal{A}} q_1$ and $c \to^{\not\in *}_{\mathcal{A}} q_1$ though $a \neq_{E_{\mathcal{B}}} c$. As a consequence, if we complete $\mathcal{A}$ with $\mathcal{R}$ and $E_{\mathcal{B}}$, we obtain an automaton that roughly approximates $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. In particular, this automaton recognizes the term c that is not reachable by rewriting the initial language $\mathcal{L}(\mathcal{A}) = \{f(a), f(c)\}$ with $\mathcal{R}$ (nor by rewriting with $\mathcal{R}/E_{\mathcal{B}}$). The completed automaton can be obtained using the Timbuk tool [13]:

States `q0 q1` **Final States** `q0` **Transitions** `c->q1 a->q1 c->q0 f(q0)->q0 f(q1)->q0 a->q0`

When E is an empty set of equations, it is possible to transform $\mathcal{R}$ and $\mathcal{A}$ to have a $\mathcal{R}/E$-coherent initial completion setting [12]. However, such a transformation is not usable, in general, when $E \neq \emptyset$. Here, for a given E possibly not empty, we propose to transform $\mathcal{A}$ so that it becomes $\mathcal{R}/E$-coherent: we build the product between $\mathcal{A}$ and either $\mathbf{MN}(E)$ or $\mathbf{E2A}(\overrightarrow{E})$. We recall the definition of product automata and we show that the product is $\mathcal{R}/E$-coherent.

▶ **Definition 31** (Product automaton [6])**.** Let $\mathcal{A} = (\mathcal{F}, Q, Q_F, \Delta_{\mathcal{A}})$ and $\mathcal{B} = (\mathcal{F}, P, P_F, \Delta_{\mathcal{B}})$ be automata. The product of $\mathcal{A}$ and $\mathcal{B}$ is $\mathcal{A} \times \mathcal{B} = (\mathcal{F}, Q \times P, Q_F \times P_F, \Delta)$ where $\Delta = \{f((q_1, p_1), \ldots, (q_k, p_k)) \to (q', p') \mid f(q_1, \ldots, q_k) \to q' \in \Delta_{\mathcal{A}} \text{ and } f(p_1, \ldots, p_k) \to p' \in \Delta_{\mathcal{B}}\}$.

▶ **Theorem 32** (Generalized Upper Bound). *Let $\mathcal{R}$ be a left-linear TRS, $\mathcal{A}$ an epsilon-free automaton, and E a set of ground equations such that $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite. If $\mathcal{B} = \mathbf{MN}(E)$ and $\mathbb{A} = \mathcal{A} \times \mathcal{B}$ then for any $i \in \mathbb{N}$: $\mathcal{L}(\mathbb{A}^i_{\mathcal{R},E}) \subseteq \mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$.*

Proof. Since $\mathcal{L}(\mathbb{A}) = \mathcal{L}(\mathcal{A} \times \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{B})$ and $\mathcal{L}(\mathcal{B}) = \mathcal{T}(\mathcal{F})$, we get that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbb{A})$. Since both $\mathcal{A}$ and $\mathcal{B}$ are epsilon-free, so is $\mathcal{B}$. Thus, to prove $\mathcal{R}/E$-coherence of $\mathbb{A}$, we only have to prove that for all states q of $\mathbb{A}$ and for all two terms $s, t \in \mathcal{T}(\mathcal{F})$ such that (1) $s \to^{\setminus *}_{\mathbb{A}} q$ and (2) $t \to^{\setminus *}_{\mathbb{A}} q$ then $s =_E t$. Since $\mathbb{A}$ is a product automaton, q is a pair of the form (q_1, q_2) where $q_1 \in \mathcal{A}$ and $q_2 \in \mathcal{B}$. From (1) and (2) we can deduce that $s \to^{\setminus *}_{\mathcal{B}} q_2$ and $t \to^{\setminus *}_{\mathcal{B}} q_2$. Then, using Lemma 11, we get $s =_E t$. Thus $\mathbb{A}$ is $\mathcal{R}/E$-coherent and from Theorem 8, we get that $\mathcal{L}(\mathbb{A}^i_{\mathcal{R},E}) \subseteq \mathcal{R}^*_E(\mathcal{L}(\mathbb{A}))$. The fact that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbb{A})$ ends the proof. ◀

▶ **Example 33.** Starting from Example 30, we can build the product between $\mathcal{A}$ and the automaton $\mathcal{B}$ found in Example 13. In $\mathcal{A} \times \mathcal{B}$, a and c are recognized by two different states, avoiding the $\mathcal{R}/E$-coherence problem of Example 30. The $\setminus$-reduced product $\mathbb{A} = \mathcal{A} \times \mathcal{B}$ (where product states are renamed) is the automaton with $\mathcal{Q}_f = \{q_2\}$ and $\Delta = \{c \to q_0, a \to q_1, f(q_0) \to q_2, f(q_1) \to q_2\}$. Running Timbuk on $\mathbb{A}$, $\mathcal{R}$, and $E_{\mathcal{B}}$, we obtain $\mathbb{A}^*_{\mathcal{R},E}$ whose precision is now bounded by $\mathcal{R}^*_{E_{\mathcal{B}}}(\mathcal{L}(\mathcal{A}))$ and does not recognize c in a final state:

```
States q0 q1 q2 Final States q0 Transitions a->q1 f(q0)->q0 f(q1)->q0 f(q2)->q0
a->q0 c->q2
```

Now, we have hints to define equations for completion. For instance, it is possible to start from an automaton $\mathcal{B}$ defining a rough approximation of the target language and build $E = \mathbf{A2E}(\mathcal{B})$. Then, we complete $\mathbb{A} = \mathcal{A} \times \mathcal{B}$ with $\mathcal{R}$ and E and obtain a tree automaton $\mathbb{A}^*_{\mathcal{R},E}$ whose precision is better or equal to $\mathcal{B}$. The set $\mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$ acts as a safeguard for completion (see Figure 1). In particular, terms of $\mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$ may not belong to $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E})$. This is the case in Example 33, where the term b belongs to $\mathcal{R}^*_{E_{\mathcal{B}}}(\mathcal{L}(\mathcal{A}))$ but not to $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E})$. For this result to be usable in practice, we still need to know if E always exists (next Section) and to generate a satisfactory E (Section 8).

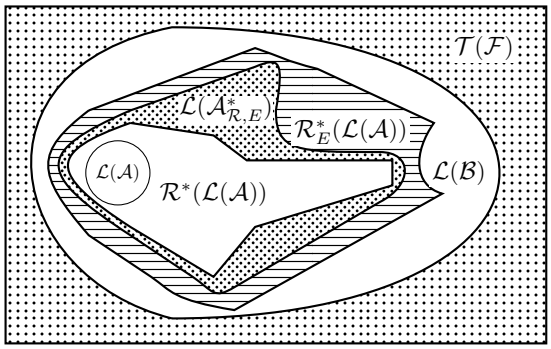


Figure 1 The Generalized Upper Bound theorem (precision of completion).

7 Completeness Theorems

In this section, we prove two completeness theorems on completion. The first theorem states that if the set of reachable terms can be over-approximated by a regular language $\mathscr{L}$, then we can find it using equational completion. The second theorem states that if the

set of reachable terms is regular then completion can build it. Since the upper-bound of completion depends on $\mathcal{R}^*_E$, we first need a lemma showing that if E is built from $\mathscr{L}$ then $\mathcal{R}^*_E$ upper-bounded by $\mathscr{L}$.

▶ **Lemma 34.** *Let $\mathcal{R}$ be a TRS over $\mathcal{F}$, $S \subseteq \mathcal{T}(\mathcal{F})$, and $\mathcal{B}$ an* **RDFC** *automaton such that $\mathcal{L}(\mathcal{B}) \supseteq \mathcal{R}^*(S)$ and $\mathcal{L}(\mathcal{B})$ is $\mathcal{R}$-closed. If $E_\mathcal{B} = \mathbf{A2E}(\mathcal{B})$ then $\mathcal{R}^*_{E_\mathcal{B}}(S) \subseteq \mathcal{L}(\mathcal{B})$.*

Proof. We prove that for all natural number $k >= 0$, if $s \in S$ and $s \to^k_{\mathcal{R}/E_\mathcal{B}} t$ then $t \in \mathcal{L}(\mathcal{B})$ where $\to^k_{\mathcal{R}/E_\mathcal{B}}$ denotes k steps of rewriting by $\mathcal{R}$ modulo $E_\mathcal{B}$. By induction on k. If $k = 0$ then $s =_{E_\mathcal{B}} t$. Using Lemma 23 on $s =_{E_\mathcal{B}} t$, we get that there exists a state q of $\mathcal{B}$ such that $s \to^*_\mathcal{B} q$ and $t \to^*_\mathcal{B} q$. Since $s \in S$ and $S \subseteq \mathcal{L}(\mathcal{B})$ there exists a final state q_f of $\mathcal{B}$ such that $s \to^*_\mathcal{B} q_f$. Since $\mathcal{B}$ is deterministic we obtain that $q = q_f$. Thus t is recognized by $\mathcal{B}$. For the inductive case, we assume that the property is true for a given k and we show that it is true for $k + 1$. Let $s \to^{k+1}_{\mathcal{R}/E} t$, i.e., we have terms s', s'', and t' such that $s \to^k_{\mathcal{R}/E} s' =_{E_\mathcal{B}} s'' \to_\mathcal{R} t' =_{E_\mathcal{B}} t$. Using the induction hypothesis, we get that s' is recognized by $\mathcal{B}$. Since $\mathcal{L}(\mathcal{B})$ is $\mathcal{R}$-closed, we know that t' is also recognized by $\mathcal{B}$. Thus, there exists a final state q_f such that $t' \to^*_\mathcal{B} q_f$. Finally, as above, applying Lemma 23 on the fact that $t' \to^*_\mathcal{B} q_f$ and $t' =_{E_\mathcal{B}} t$ gives us that $t \to^*_\mathcal{B} q_f$. ◀

Next example shows that the $\mathcal{R}$-closed assumption on $\mathscr{L}$ is necessary for the lemma to hold.

▶ **Example 35.** Let $\mathcal{F} = \{a : 0, b : 0, c : 0, d : 0\}$, $S = \{a\}$, $\mathcal{R} = \{a \to b, c \to d\}$, and $\mathscr{L} = \{a, b, c\}$ where $\mathscr{L} \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ but $\mathscr{L}$ is not $\mathcal{R}$-closed. A possible **RDFC** automaton $\mathcal{B}$, s.t. $\mathcal{L}(\mathcal{B}) = \mathscr{L}$, has a unique final state q and transitions $\{a \to q, b \to q, c \to q\}$. Thus $E_\mathcal{B} = \mathbf{A2E}(\mathcal{B})$ will include the equation $b = c$. Finally $\mathcal{R}^*_{E_\mathcal{B}}(S) = \{a, b, c, d\} \not\subseteq \mathscr{L}$.

▶ **Theorem 36** (Completeness)**.** *Let $\mathcal{A}$ be a reduced epsilon-free tree automaton and $\mathcal{R}$ a left-linear TRS. Let $\mathcal{T}(\mathcal{F}) \supseteq \mathscr{L} \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. If $\mathscr{L}$ is regular and $\mathcal{R}$-closed then there exists a set of ground equations E such that $\mathbb{A} = \mathcal{A} \times \mathbf{MN}(E)$, $\mathbb{A}^*_{\mathcal{R},E}$ exists and $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathscr{L}$.*

Proof. Since $\mathscr{L}$ is regular, we know that there exists an **RDFC** tree automaton, say $\mathcal{B}$, recognizing $\mathscr{L}$. From $\mathcal{B}$ we can infer $E_\mathcal{B} = \mathbf{A2E}(\mathcal{B})$ and then use completion to compute reachable terms. From Theorem 26, we know that completion of the automaton $\mathcal{A}$ with $\mathcal{R}$ and the set of equations $E_\mathcal{B}$ always terminates on a tree automaton $\mathcal{A}^*_{\mathcal{R},E_\mathcal{B}}$. From Theorem 8, we know that $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E_\mathcal{B}}) \subseteq \mathcal{R}^*_{E_\mathcal{B}}(\mathcal{L}(\mathcal{A}))$ provided that $\mathcal{A}$ is $\mathcal{R}/E_\mathcal{B}$-coherent. To enforce $\mathcal{R}/E_\mathcal{B}$-coherence of $\mathcal{A}$, we apply the transformation presented in Section 6. Let $\mathbb{A} = \mathcal{A} \times \mathbf{MN}(E_\mathcal{B})$. Note that since $E_\mathcal{B}$ is obtained by using the **A2E** transformation, $\mathcal{T}(\mathcal{F})/_{=_E}$ is finite (Corollary 24) and since equations of $E_\mathcal{B}$ are ground, $=_{E_\mathcal{B}}$ is decidable. The resulting automaton $\mathbb{A}$ is $\mathcal{R}/E_\mathcal{B}$-coherent. Besides, Theorem 26 also applies to $\mathbb{A}$. Thus, completion of $\mathbb{A}$ with $\mathcal{R}$ and $E_\mathcal{B}$ always ends on an automaton $\mathbb{A}^*_{\mathcal{R},E_\mathcal{B}}$. The automaton $\mathbb{A}^*_{\mathcal{R},E_\mathcal{B}}$ satisfies $\mathcal{R}^*(\mathcal{L}(\mathbb{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E_\mathcal{B}})$ (by Theorem 5) and $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E_\mathcal{B}}) \subseteq \mathcal{R}^*_{E_\mathcal{B}}(\mathcal{L}(\mathbb{A}))$ (by Theorem 32). Since $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathbb{A})$, we have $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E_\mathcal{B}})$ and $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E_\mathcal{B}}) \subseteq \mathcal{R}^*_{E_\mathcal{B}}(\mathcal{L}(\mathcal{A}))$. With Lemma 34, we get that $\mathcal{R}^*_{E_\mathcal{B}}(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{B}) = \mathscr{L}$. ◀

Note that, in general we do not have $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \supseteq \mathscr{L}$ because $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E})$ can be more precise than $\mathscr{L}$. However, this is true when $\mathscr{L} = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$, as we show is the next theorem.

▶ **Corollary 37.** *Let $\mathcal{R}$ be a TRS over $\mathcal{F}$, $S \subseteq \mathcal{T}(\mathcal{F})$, and $\mathcal{B}$ an* **RDFC** *automaton such that $\mathcal{L}(\mathcal{B}) = \mathcal{R}^*(S)$. If $E_\mathcal{B} = \mathbf{A2E}(\mathcal{B})$ then $\mathcal{R}^*_{E_\mathcal{B}}(S) = \mathcal{R}^*(S)$.*

Proof. Note that since $\mathcal{L}(\mathcal{B}) = \mathcal{R}^*(S)$ it is $\mathcal{R}$-closed. Then, using Lemma 34, we get that $\mathcal{R}^*(S) \subseteq \mathcal{R}^*_{E_\mathcal{B}}(S) \subseteq \mathcal{L}(\mathcal{B})$. Since $\mathcal{L}(\mathcal{B}) = \mathcal{R}^*(S)$, we get the result. ◀

▶ **Theorem 38** (Completeness for regularity preserving TRSs). *Let $\mathcal{A}$ be a reduced epsilon-free tree automaton and $\mathcal{R}$ a left-linear TRS. If $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ is regular then it is possible to compute a tree automaton recognizing $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ by equational tree automata completion.*

Proof. Let $\mathscr{L} = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. It is $\mathcal{R}$-closed. By assumption, it is also regular. Thus, we can apply Theorem 36 to get that there exists a set of equations E and a tree automaton $\mathbb{A} = \mathcal{A} \times \mathbf{MN}(E)$ such that $\mathbb{A}^*_{\mathcal{R},E}$ exists and $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathscr{L}$. Since $\mathscr{L} = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$, we get $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) = \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. ◀

Thus, completion is complete w.r.t. *all left-linear TRS classes preserving regularity.*

8 Application of the Completeness Theorem

In the previous section, we have proved that if there exists a regular over-approximation of sets of reachable terms, then we can build it using completion. Now, we show how to take advantage of this theorem to automatically verify safety properties on programs. Given an initial regular language S and a program represented by a TRS $\mathcal{R}$, we can prove that the program never reaches terms in a set Bad by checking that there exists a regular over-approximation $\mathscr{L} \supseteq \mathcal{R}^*(S)$ such that $\mathscr{L} \cap Bad = \emptyset$. This technique has been used to verify cryptographic protocols [15], Java programs [3] and Functional Programs [12, 14]. Theorem 36 ensures that, if there exists an $\mathcal{R}$-closed regular approximation $\mathscr{L}$ such that $\mathscr{L} \cap Bad = \emptyset$, then we can build it (or under-approximate it) using completion and an appropriate set E. To explore all the possible E, it is enough to explore $G_\mathcal{F}(k)$ with $k \in \mathbb{N}^*$.

▶ **Definition 39** (Generated Equations for $\mathcal{F}$ and $k \in \mathbb{N}^*$). Let $\mathbb{B}(k)$ be the set of all possible **RDFC** tree automata on $\mathcal{F}$ with exactly k states. The set of *generated equations* of size k is $G_\mathcal{F}(k) = \{E \mid \mathcal{B} \in \mathbb{B}(k) \text{ and } E = \mathbf{A2E}(\mathcal{B})\}$.

The semi-algorithm to prove that $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \cap Bad = \emptyset$ works as follows: (a) We start from $k = 1$, (b) we generate $G_\mathcal{F}(k)$, (c) we try completion with $\mathcal{A}$, $\mathcal{R}$ and all $E \in G_\mathcal{F}(k)$ (completion terminates with all those E, Theorem 26). If $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \cap Bad = \emptyset$ for one E, we are done. Otherwise if $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \cap Bad \neq \emptyset$ for all $E \in G_\mathcal{F}(k)$, we increase k and go back to step (b). If there exists a regular over-approximation $\mathscr{L} \supseteq \mathcal{R}^*(S)$, this algorithm eventually reaches a tree automaton $\mathcal{B}$ such that $\mathcal{L}(\mathcal{B}) = \mathscr{L}$, $E = \mathbf{A2E}(\mathcal{B}$, and by Theorem 36, we know that $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \subseteq \mathscr{L}$. Finally, since $\mathscr{L} \cap Bad = \emptyset$, we have $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \cap Bad = \emptyset$.

For general TRSs, we can enumerate all equation sets but the search space is huge. When the TRS $\mathcal{R}$ encodes a functional program, we can restrict the search space to equation sets of the form $E = E_\mathcal{R} \cup E_r \cup E_\mathcal{C}$ [12], where $E_\mathcal{R}$ and E_r are fixed and $E_\mathcal{C}$ only ranges over $\textsc{Irr}(\mathcal{R})$. If program's functions are complete, $\textsc{Irr}(\mathcal{R})$ is the set of are constructor terms, i.e., terms containing no function call. The set $\mathcal{F}$ can be separated into a set of *defined symbols* $\mathcal{D} = \{f \mid \exists l \to r \in \mathcal{R} \text{ s.t. } \mathscr{R}oot(l) = f\}$ and *constructor symbols* $\mathcal{C} = \mathcal{F} \setminus \mathcal{D}$.

▶ **Definition 40** (E_r). For a given set of symbols $\mathcal{F}$, $E_r = \{f(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) \mid f \in \mathcal{F}$, and arity of f is $n\}$, where $x_1 \ldots x_n$ are pairwise distinct variables.

▶ **Definition 41** ($E_\mathcal{R}$). Let $\mathcal{R}$ be a TRS, the set of $\mathcal{R}$-equations is $E_\mathcal{R} = \{l = r \mid l \to r \in \mathcal{R}\}$.

▶ **Definition 42** ($E_{\mathcal{C}}$ contracting equations for $\mathcal{T}(\mathcal{C})$)**.** A set of equations is contracting for $\mathcal{T}(\mathcal{C})$, denoted by $E_{\mathcal{C}}$, if all equations of $E_{\mathcal{C}}$ are of the form $u = u|_p$ with u a linear term of $\mathcal{T}(\mathcal{C}, \mathcal{X})$, $p \neq \lambda$, and $\textsc{Irr}(\overrightarrow{E_{\mathcal{C}}})$ the set of terms of $\mathcal{T}(\mathcal{C})$ irreducible by $\overrightarrow{E_{\mathcal{C}}}$ is finite, where $\overrightarrow{E_{\mathcal{C}}} = \{u \to u|_p \mid u = u|_p \in E_{\mathcal{C}}\}$.

Completion is terminating if $E = E_{\mathcal{R}} \cup E_r \cup E_{\mathcal{C}}$ and $\mathcal{R}$ encodes a functional program that is terminating, complete, and either first order [12] or higher-order [14]. Now, our objective is to define a completeness theorem for TRSs encoding those programs. Since E contains $E_{\mathcal{R}}$, all completed automata $\mathcal{A}^*_{\mathcal{R},E}$ will be $\mathcal{R}$-closed because $s \to^*_{\mathcal{A}^*_{\mathcal{R},E}} q$, $s \to_{\mathcal{R}} t$, $t \to^*_{\mathcal{A}^*_{\mathcal{R},E}} q'$ implies that $s =_E t$ and $q = q'$ because $\mathcal{A}^*_{\mathcal{R},E}$ is simplified w.r.t. E. Thus, the completeness theorem says that if there exists an $\mathcal{R}$-closed automaton $\mathcal{B}$ s.t. $\mathcal{L}(\mathcal{B}) \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ then there exists $E_{\mathcal{C}}$ such that $E = E_{\mathcal{R}} \cup E_r \cup E_{\mathcal{C}}$ and $\mathcal{L}(\mathcal{A}^*_{\mathcal{R},E}) \subseteq \mathcal{L}(\mathcal{B})$. To prove such a theorem, we need to explain how to construct a satisfying $E_{\mathcal{C}}$ from $\mathcal{B}$. We propose to project $\mathcal{B}$ on $\mathcal{C}$ (denoted by $\mathcal{B}/\mathcal{C}$), produce equations from $\mathcal{B}/\mathcal{C}$ with **A2E**, and finally filter out all equations that are not of the form $u = u|_p$ (this is function ct).

▶ **Definition 43** (Automaton projection on $\mathcal{C}$)**.** Let $\mathcal{B} = \langle \mathcal{F}, \mathcal{Q}, \mathcal{Q}_f, \Delta \rangle$ be an epsilon free tree automaton. The automaton $\mathcal{B}/\mathcal{C}$ is the tree automaton $\langle \mathcal{C}, \mathcal{Q}_{\mathcal{C}}, \mathcal{Q}_f \cap \mathcal{Q}_{\mathcal{C}}, \Delta_{\mathcal{C}} \rangle$ where $\Delta_{\mathcal{C}} = \{s \to q \mid s \to q \in \Delta \wedge \mathscr{R}oot(s) \in \mathcal{C}\}$ and $\mathcal{Q}_{\mathcal{C}}$ is the set of states occurring in the right-hand side of transitions of $\Delta_{\mathcal{C}}$.

Note that $\mathcal{L}(\mathcal{B}/\mathcal{C}) = \mathcal{L}(\mathcal{B}) \cap \mathcal{T}(\mathcal{C})$ and if $\mathcal{B}$ is **RDFC** so is $\mathcal{B}/\mathcal{C}$. In particular, if $\mathcal{B}$ is complete for $\mathcal{F}$, $\mathcal{B}/\mathcal{C}$ is complete for $\mathcal{C}$.

▶ **Definition 44.** Given a set of equations E, $ct(E) = \{l = r \in E \mid r = l|_p \text{ and } p \neq \lambda\}$.

In the following, we show that $E = ct(\mathbf{A2E}(\mathcal{B}))$ is a contracting set of equations, provided that $\mathcal{B}$ is **RDFC**. In particular, we show that $\textsc{Irr}(\overrightarrow{E})$ is finite. Recall that $\overrightarrow{E}$ denotes the TRS where all equations $l = r$ of E are oriented so that r is a strict subterm of l.

▶ **Lemma 45.** *Let $\mathcal{B}$ be an* **RDFC** *automaton on $\mathcal{C}$ and $E = ct(\mathbf{A2E}(\mathcal{B}))$. For all $t \in \mathcal{T}(\mathcal{C})$ and $q \in \mathcal{B}$, t can be rewritten by $\overrightarrow{E}$ iff $t \notin [\![q]\!]_{\mathcal{B}}$.*

Proof. By induction on the height of t.

- If $|t| = 1$, then by Definition 17, t is necessarily a state representative of q, i.e., $t \in [\![q]\!]_{\mathcal{B}}$. By definition of ct and $\overrightarrow{E}$, we know that $\overrightarrow{E}$ does not contain any rule with a constant on the left-hand side. Thus t cannot be rewritten by $\overrightarrow{E}$.
- Assume that the property is true for terms of height lesser than k. Let $t = f(t_1, \ldots, t_n)$ be a term of height k.
  - We prove that if $t \in [\![q]\!]_{\mathcal{B}}$ then t cannot be rewritten by $\overrightarrow{E}$. We make a proof by contradiction. Assume that $t \in [\![q]\!]_{\mathcal{B}}$ and t can be rewritten by $\overrightarrow{E}$. Since $t = f(t_1, \ldots, t_n) \in [\![q]\!]_{\mathcal{B}}$ then t_i, for $1 \leq i \leq n$, are state representatives. Thus, using the induction hypothesis, we get that t_i $(1 \leq i \leq n)$ are irreducible by $\overrightarrow{E}$. Thus, the term t can only be rewritten at position λ. For t to be rewritten at position λ there should be an equation of the form $t = t|_p$ in $E = ct(\mathbf{A2E}(\mathcal{B}))$, with $p \in \mathscr{P}os(t)$ and $p \neq \lambda$. For $t = t|_p$ to belong to $\mathbf{A2E}(\mathcal{B})$, we necessarily have a transition $f(q_1, \ldots, q_n) \to q \in \mathcal{B}$ and $t|_p \in [\![q]\!]_{\mathcal{B}}$. By definition of state representatives, from $t \in [\![q]\!]_{\mathcal{B}}$ we deduce that no subterm of t can be a state representative of q. This is a contradiction with $t|_p \in [\![q]\!]_{\mathcal{B}}$.

- We prove that if $t \notin [\![q]\!]_{\mathcal{B}}$ then t can be rewritten by $\overrightarrow{E}$. From $t = f(t_1, \ldots, t_n) \to_{\mathcal{B}}^* q$, we know that there exists states $q_1, \ldots, q_n \in \mathcal{B}$ such that $f(t_1, \ldots, t_n) \to_{\mathcal{B}}^* f(q_1, \ldots, q_n) \to_{\mathcal{B}} q$, i.e., $t_i \to_{\mathcal{B}}^* q_i$ for $1 \leq i \leq n$. Applying the induction hypothesis on all the t_i's, we obtain that t_i can be rewritten by $\overrightarrow{E}$ iff $t_i \notin [\![q_i]\!]_{\mathcal{B}}$. If there exists one t_i s.t. $t_i \notin [\![q_i]\!]_{\mathcal{B}}$ then t_i can be rewritten by $\overrightarrow{E}$ and so is $t = f(t_1, \ldots, t_n)$. Thus, for all $1 \leq i \leq n$, $t_i \in [\![q_i]\!]_{\mathcal{B}}$. Besides, recall that $f(q_1, \ldots, q_n) \to q \in \mathcal{B}$ and $t = f(t_1, \ldots, t_n) \notin [\![q]\!]_{\mathcal{B}}$. By definition of state representatives, from $t \notin [\![q]\!]_{\mathcal{B}}$ we can deduce that there exists a position $p \in \mathscr{P}os(t)$ and $p \neq \lambda$ such that $t|_p \in [\![q]\!]_{\mathcal{B}}$. Since $f(q_1, \ldots, q_n) \to q \in \mathcal{B}$ and for all $1 \leq i \leq n$, $t_i \in [\![q_i]\!]_{\mathcal{B}}$, the equation $f(t_1, \ldots, t_n) = t|_p$ belongs to $\mathbf{A2E}(\mathcal{B})$. The equation also belongs to $E = ct(\mathbf{A2E}(\mathcal{B}))$. Thus $t = f(t_1, \ldots, t_n)$ can be rewritten by $\overrightarrow{E}$.

◀

▶ **Lemma 46.** *If $\mathcal{B}$ is an* **RDFC** *automaton on $\mathcal{C}$ and $E = ct(\mathbf{A2E}(\mathcal{B}))$, then* $\textsc{Irr}(\overrightarrow{E})$ *is finite and E is contracting for $\mathcal{T}(\mathcal{C})$.*

Proof. If $\textsc{Irr}(\overrightarrow{E})$ is finite, then we trivially have that E is contracting for $\mathcal{T}(\mathcal{C})$. We prove that $\textsc{Irr}(\overrightarrow{E})$ is finite by contradiction. Assume that $\textsc{Irr}(\overrightarrow{E})$ is infinite. Since $\overrightarrow{E}$ is left-linear, we know that $\textsc{Irr}(\overrightarrow{E})$ is regular. Thus, we know that there exists an automaton $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = \textsc{Irr}(\overrightarrow{E})$. Let $\mathcal{A} = \langle \mathcal{C}, \mathcal{Q}^{\mathcal{A}}, \mathcal{Q}_f^{\mathcal{A}}, \Delta_{\mathcal{A}} \rangle$ and $\mathcal{B} = \langle \mathcal{C}, \mathcal{Q}^{\mathcal{B}}, \mathcal{Q}_f^{\mathcal{B}}, \Delta_{\mathcal{B}} \rangle$. Let $\mathcal{D}$ be the automaton $\mathcal{A} \times \mathcal{B}$ where the set of final states of $\mathcal{D}$, $\mathcal{Q}_f^{\mathcal{D}}$ is $\mathcal{Q}_f^{\mathcal{A}} \times \mathcal{Q}^{\mathcal{B}}$. Since $\mathcal{B}$ is complete, we thus have $\mathcal{L}(\mathcal{D}) = \mathcal{L}(\mathcal{A})$. Let $q_f \in \mathcal{Q}_f^{\mathcal{D}}$ and a ground term $t \in \mathcal{T}(\mathcal{C})$ such that $t \to_{\Delta}^* q_f$ and t does not belong to state representatives of $\mathcal{B}$. We know that such a t exists because the set of state representatives of $\mathcal{B}$ is finite (Lemma 19) and $\mathcal{L}(\mathcal{D})$ is infinite. Since $\mathcal{D} = \mathcal{A} \times \mathcal{B}$, we know that q_f is of the form $(q_f^{\mathcal{A}}, q^{\mathcal{B}})$ with $q_f^{\mathcal{A}} \in \mathcal{Q}_f^{\mathcal{A}}$ and $q^{\mathcal{B}} \in \mathcal{Q}^{\mathcal{B}}$, and we necessarily have (a) $t \to_{\mathcal{A}}^* q_f^{\mathcal{A}}$ and (b) $t \to_{\mathcal{B}}^* q^{\mathcal{B}}$. The fact that t is not a state representative of any state of $\mathcal{B}$ entails that, in particular, $t \notin [\![q^{\mathcal{B}}]\!]_{\mathcal{B}}$. Finally, with (b) and $t \notin [\![q^{\mathcal{B}}]\!]_{\mathcal{B}}$, we can apply Lemma 45 and get that t can be rewritten with $\overrightarrow{E}$. This contradicts (a) because $t \to_{\mathcal{A}}^* q_f^{\mathcal{A}}$ implies that $t \in \textsc{Irr}(\overrightarrow{E})$. ◀

The above lemma implies that any regular language (on $\mathcal{C}$) can be defined using a set of contracting equations on $\mathcal{T}(\mathcal{C})$.

▶ **Lemma 47.** *For a TRS $\mathcal{R}$ and an automaton $\mathcal{B}$ on $\mathcal{F}$, if $\mathcal{B}$ is* **RDFC** *and $\mathcal{R}$-closed and $E_{\mathcal{B}} = \mathbf{A2E}(\mathcal{B})$, $E_{\mathcal{C}} = ct(\mathbf{A2E}(\mathcal{B}/\mathcal{C}))$, and $E = E_{\mathcal{R}} \cup E_r \cup E_{\mathcal{C}}$ then $=_E \subseteq =_{E_{\mathcal{B}}}$.*

Proof. We prove that if $s =_E t$ then $s =_{E_{\mathcal{B}}} t$ by induction on the number $k \in \mathbb{N}$ of equational steps necessary to have $s =_E t$. If $k = 0$ then $s = t$ and this is trivially true for $=_{E_{\mathcal{B}}}$. For the inductive case, we have $s =_E s_2 =_E \ldots =_E s_k =_E t$. We can apply the induction hypothesis to get $s =_{E_{\mathcal{B}}} s_1 =_{E_{\mathcal{B}}} \ldots =_{E_{\mathcal{B}}} s_k$. To prove $s_k =_{E_{\mathcal{B}}} t$ from $s_k =_E t$, we do a proof by case on the equation used in the step $s_k =_E t$.

- If $s_k =_{E_{\mathcal{C}}} t$, because of a equation $u = v$ in $E_{\mathcal{C}}$, then we know that $u = v$ necessarily appears in $E_{\mathcal{B}}$. This is due to the fact that the set of transitions of $\mathcal{B}/\mathcal{C}$ is included in the set of transitions of $\mathcal{B}$ and thus $E_{\mathcal{C}} = ct(\mathbf{A2E}(\mathcal{B}/\mathcal{C}))$ is included in $E_{\mathcal{B}} = \mathbf{A2E}(\mathcal{B})$;
- If $s_k =_{E_r} t$, then the equation is of the form $u = u$ and $s_k = t$ which implies $s_k =_{E_{\mathcal{B}}} t$;
- If $s_k =_{E_{\mathcal{R}}} t$ then we either have $s_k \to_{\mathcal{R}} t$ or $t \to_{\mathcal{R}} s_k$. Since $\mathcal{B}$ is **RDFC** we know that there exists a state $q \in \mathcal{B}$ such that $s_k \to_{\mathcal{B}}^* q$. Furthermore, since $\mathcal{B}$ is $\mathcal{R}$-closed, we know that $t \to_{\mathcal{B}}^* q$. By Lemma 23, we get that $s_k =_{E_{\mathcal{B}}} t$.

◀

▶ **Theorem 48** ($E_\mathcal{R} \cup E_r \cup E_\mathcal{C}$ covers all $\mathcal{R}$-closed approximation automata)**.** *Let $\mathcal{R}$ be a left-linear TRS and $\mathcal{A}$ a reduced and epsilon-free tree automaton on $\mathcal{F}$. Let $\mathcal{B}$ be an $\mathcal{R}$-closed* **RDFC** *tree automaton such that $\mathcal{L}(\mathcal{B}) \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. Let $E_\mathcal{C} = ct(\mathbf{A2E}(\mathcal{B}/\mathcal{C}))$, $E = E_\mathcal{C} \cup E_\mathcal{R} \cup E_r$, and $\mathbb{A} = \mathcal{A} \times \mathbf{MN}(E)$. If $\mathbb{A}^*_{\mathcal{R},E}$ exists then $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathcal{L}(\mathcal{B})$.*

Proof. The fact that $\mathcal{R}^*(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathbb{A}^*_{\mathcal{R},E})$ is ensured by Theorem 5. Using the Generalized Upper Bound theorem (Theorem 32), we deduce that (1) $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathcal{R}^*_E(\mathcal{L}(\mathcal{A}))$. From Lemma 47, we know that $=_E \subseteq =_{E_\mathcal{B}}$ and thus that (2) $\mathcal{R}^*_E(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{R}^*_{E_\mathcal{B}}(\mathcal{L}(\mathcal{A}))$. Besides, since $\mathcal{B}$ is $\mathcal{R}$-closed, $\mathcal{L}(\mathcal{B})$ is $\mathcal{R}$-closed and we can use Lemma 34 to get that (3) $\mathcal{R}^*_{E_\mathcal{B}}(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{B})$. Finally, using transitivity of $\subseteq$ on (1), (2) and (3) we get $\mathcal{L}(\mathbb{A}^*_{\mathcal{R},E}) \subseteq \mathcal{L}(\mathcal{B})$. ◀

Note that, for functional programs classes of [12] and [14], since $E_\mathcal{C} = ct(\mathbf{A2E}(\mathcal{B}/\mathcal{C}))$ is contracting (Lemma 46), $\mathbb{A}^*_{\mathcal{R},E}$ always exists. Thus, if there exists an $\mathcal{R}$-closed tree automaton $\mathcal{B}$ such that $\mathcal{L}(\mathcal{B}) \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ and $\mathcal{L}(\mathcal{B}) \cap Bad = \emptyset$, it is enough to enumerate all possible $E = E_\mathcal{R} \cup E_r \cup E_\mathcal{C}$ to find it. Since $E_\mathcal{R}$ and E_r are fixed, it is enough to enumerate all possible $E_\mathcal{C}$ on $\mathcal{C}$ using Definition 39.

▶ **Example 49.** Let $\mathcal{C} = \{0 : 0, s : 1\}$. For $k = 1$, there is only one **RDFC** automaton with 1 state. Its transitions are $\{s(q_0) \to q_0, 0 \to q_0\}$. Thus, $G_\mathcal{C}(1) = \{\{s(0) = 0\}\}$. For $k = 2$ there are 2 **RDFC** automata : one with transitions $\{0 \to q_0, s(q_0) \to q_1, s(q_1) \to q_1\}$ and the other with transitions $\{0 \to q_0, s(q_0) \to q_1, s(q_1) \to q_0\}$. Thus, $G_\mathcal{C}(2) = \{\{s(s(0)) = s(0)\}, \{s(s(0)) = 0, s(s(s(0))) = s(0)\}\}$.

We implemented this in Timbuk and used it verify safety properties of several first-order functions on lists and trees, higher-order functions [14]: *map*, *filter*, *exists*, *forall*, $foldRight$, $foldLeft$ as well as higher-order sorting functions parameterized by an ordering function. Most of examples are taken from [22, 20], except the examples of functions manipulating trees. Contracting equations used in [14] contain variables and are generated from test sets. Here, we generate ground contracting equations $E_\mathcal{C}$ as shown above and use $E = E_\mathcal{R} \cup E_r \cup E_\mathcal{C}$ for completion. We transform the initial automaton $\mathcal{A}$ into $\mathbb{A}$ as in Theorem 32. The approximation is, thus, upper-bounded by $\mathcal{R}^*_E$ and we can benefit from the coverage guarantee of Theorem 48. On all the examples of [14], we managed to do the same proofs (or find the counter-examples, see [14]), in a faster way. Full experiments can be found here: `http://people.irisa.fr/Thomas.Genet/timbuk/funExperiments/`. Those experiments show that, with equation enumeration, the general tree automata completion algorithm becomes powerful enough to efficiently carry out safety proofs on first-order and higher-order functional programs. In [22, 20], equivalent proofs require higher-order model-checking tools that are specialized for this task.

9 Conclusion and perspectives

Tree automata completion is known to cover many TRS classes preserving regularity [10, 12]. For some other classes, such as the linear subclass of [8], the question was still open. We established that, for all those classes (including those not known yet), given $\mathcal{A}$ and $\mathcal{R}$, there exists a set of equations E such that $\mathcal{A}^*_{\mathcal{R},E}$ recognizes $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$. We proved a similar theorem (Theorem 36) for the approximated case. If there exists a regular $\mathcal{R}$-closed approximation $\mathscr{L}$ such that $\mathscr{L} \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ then there exists a set of equations E such that $\mathcal{A}^*_{\mathcal{R},E}$ recognizes, or under-approximates, $\mathscr{L}$. Since building $\mathcal{R}^*(\mathcal{L}(\mathcal{A}))$ or guessing

an over-approximation $\mathscr{L}$ is impossible in general, inferring E from $\mathcal{A}$ and $\mathcal{R}$ is impossible. Nevertheless, this paper presents a technique to enumerate sets of equations E to search for a completed automaton satisfying a particular property. Theorem 29 proves that completion with those E will always terminate. Theorem 32 ensures that the completed automaton will be as precise as possible w.r.t. $\mathcal{R}/E$. Finally, Theorem 36 shows that if a regular approximation satisfying the property exists, it will be found by enumerating the sets E and running completion.

On functional programs, Theorem 48 shows that enumeration can be restricted to sets of equations on constructor symbols. This makes enumeration efficient enough to automatically verify properties on first-order and higher programs. Our experiments with this approach shows that it can prove state-of-the-art regular properties on first-order on higher-order programs. The completeness Theorem for functional programs ranges over $\mathcal{R}$-closed **RDFC** approximation automata. However, there exist $\mathcal{R}$-closed approximations that are not recognized by an $\mathcal{R}$-closed **RDFC** tree automaton.

▶ **Example 50.** Let $\mathcal{F} = \{f : 1, a : 0, b : 0\}$, $\mathcal{R} = \{a \to b\}$ and $\mathscr{L} = \{f(b), a, b\}$. The language $\mathscr{L}$ is $\mathcal{R}$-closed and regular. There exists no $\mathcal{R}$-closed **RDFC** tree automaton recognizing $\mathscr{L}$. In any $\mathcal{R}$-closed **RDFC** tree automaton, a and b needs to be recognized by the same state, say q, and thus $f(b)$ needs to be recognized using a transition $f(q) \to q_f$ where q_f is final. Thus, this automaton recognizes $f(a)$ which does not belong to $\mathscr{L}$.

Such approximations are thus out of the scope of Theorem 48, and cannot be found by enumerating $E_{\mathcal{C}}$. As explained in Section 8, since E contains $E_{\mathcal{R}}$, the completed automata are $\mathcal{R}$-closed. We think that some interesting approximations may be recognized by non-$\mathcal{R}$-closed automata. For instance, during our experiments, we succeeded in proving the sortedness property on the insertion sort but it timed out on the merge sort. However, when using the $E_{\mathcal{C}}$ set generated for the insertion sort and E_r and $E_{\mathcal{R}}$ for merge sort, completion succeeds if we remove some equations from $E_{\mathcal{R}}$. We do not know if there exists an $\mathcal{R}$-closed automaton proving the property but, this experimental result shows that the smallest automaton proving the property is not $\mathcal{R}$-closed. We think that it is possible to explore the set of all possible equation sets using $E = E_r \cup E_{\mathcal{C}}$ where $E_{\mathcal{C}}$ is contracting on $\mathcal{T}(\mathcal{F})$ and to prune the search space using CounterExample Guided Abstraction Refinement. This would permit to have an efficient equation generation for general TRSs and widen its applicability to non-terminating functional programs, cryptographic protocols, etc. This is ongoing work.

Another perspective is to extend those results to non-left-linear TRSs. Dealing with regular languages and non-left-linear rules is known to be more challenging than the left-linear case [24, 2, 9]. Nevertheless, there could be a nice surprise here. For non-left-linear TRSs, completion is known to be sound and precise as long as the completed tree automaton is kept deterministic [10]. Completion itself does not preserve determinism but, in Section 8, all the completed automata are deterministic. This is a consequence of the fact that E contains E_r (makes the automaton ϵ-deterministic) and $E_{\mathcal{R}}$ (merges all states related by an ϵ-transition). Thus, when using $E = E_{\mathcal{R}} \cup E_r \cup E_{\mathcal{C}}$, completion is likely to build over-approximations for non-left-linear TRSs. This should be investigated further.

Acknowledgments I thank Sophie Tison for discussions about the finiteness of $\mathcal{T}(\mathcal{F})/_{=_E}$ and Tristan Gillard for implementing the equation generation in Timbuk.

References

1 F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

2 Y. Boichut, R. Courbis, P.-C. Héam, and O. Kouchnarenko. Handling non left-linear rules when completing tree automata. *IJFCS*, 20(5), 2009.

3 Y. Boichut, T. Genet, T. Jensen, and L. Leroux. Rewriting Approximations for Fast Prototyping of Static Analyzers. In *RTA'07*, volume 4533 of *LNCS*, pages 48–62. Springer, 2007.

4 Y. Boichut, P.-C. Héam, and O. Kouchnarenko. Automatic Approximation for the Verification of Cryptographic Protocols. In *Proc. AVIS'2004, joint to ETAPS'04, Barcelona (Spain)*, 2004.

5 H. Comon. Sequentiality, Monadic Second-Order Logic and Tree Automata. *Inf. Comput.*, 157(1-2):25–51, 2000.

6 H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison, and M. Tommasi. Tree automata techniques and applications. `http://tata.gforge.inria.fr`, 2008.

7 H. Comon and Jean-Luc Rémy. How to characterize the language of ground normal forms. Technical Report 676, INRIA-Lorraine, 1987.

8 I. Durand and M. Sylvestre. Left-linear bounded trss are inverse recognizability preserving. In *RTA'11*, volume 10 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

9 B. Felgenhauer and R. Thiemann. Reachability Analysis with State-Compatible Automata. In *LATA'14*, volume 8370 of *LNCS*, pages 347–359. Springer, 2014.

10 G. Feuillade, T. Genet, and V. Viet Triem Tong. Reachability Analysis over Term Rewriting Systems. *Journal of Automated Reasonning*, 33 (3-4):341–383, 2004.

11 T. Genet. Decidable Approximations of Sets of Descendants and Sets of Normal Forms. In *RTA'98*, volume 1379 of *LNCS*, pages 151–165. Springer, 1998.

12 T. Genet. Termination Criteria for Tree Automata Completion. *Journal of Logical and Algebraic Methods in Programming*, 85, Issue 1, Part 1:3–33, 2016.

13 T. Genet, Y. Boichut, B. Boyer, V. Murat, and Y. Salmon. Reachability Analysis and Tree Automata Calculations. IRISA / Université de Rennes 1. `http://people.irisa.fr/Thomas.Genet/timbuk/`.

14 T. Genet, T. Haudebourg, and T. Jensen. Verifying higher-order functions with tree automata. In *FoSSaCS'18*, LNCS. Springer, 2018. To be published.

15 T. Genet and F. Klay. Rewriting for Cryptographic Protocol Verification. In *Proc. of CADE'00*, volume 1831 of *LNAI*. Springer, 2000.

16 T. Genet and R. Rusu. Equational tree automata completion. *Journal of Symbolic Computation*, 45:574–597, 2010.

17 A. Geser, D. Hofbauer, J. Waldmann, and H. Zantema. On tree automata that certify termination of left-linear term rewriting systems. In *RTA'05*, volume 3467 of *LNCS*, pages 353–367. Springer, 2005.

18 F. Jacquemard. Decidable approximations of term rewriting systems. In H. Ganzinger, editor, *Proc. 7th RTA Conf., New Brunswick (New Jersey, USA)*, pages 362–376. Springer-Verlag, 1996.

19 Dexter Kozen. On the Myhill-Nerode theorem for trees. *Bull. Europ. Assoc. Theor. Comput. Sci.*, 47:170–173, June 1992.

20 Y. Matsumoto, N. Kobayashi, and H. Unno. Automata-Based Abstraction for Automated Verification of Higher-Order Tree-Processing Programs. In *APLAS'15*, volume 9458 of *LNCS*, pages 295–312. Springer, 2015.

21 A. Middeldorp. Approximations for strategies and termination. *ENTCS*, 70(6):1–20, 2002.

22 L. Ong and S. Ramsay. Verifying higher-order functional programs with pattern-matching algebraic data types. In *POPL'11*. ACM, 2011.

23 T. Takai. A Verification Technique Using Term Rewriting Systems and Abstract Interpretation. In *RTA'04*, volume 3091 of *LNCS*, pages 119–133. Springer, 2004.

24 T. Takai, Y. Kaji, and H. Seki. Right-linear finite-path overlapping term rewriting systems effectively preserve recognizability. In *RTA'11*, volume 1833 of *LNCS*. Springer, 2000.

25 S. Tison. Finiteness of the set of E-equivalence classes is undecidable, 2010. Private communication.